CODIFICACIÓN

4.1 INTRODUCCIÓN

Desde el momento que nos planteamos el problema del manejo de la información (ej.: transmisión, recepción, almacenamiento, etc.); surge la necesidad de representar de alguna manera, esta información, que vendrá dada por eventos o sucesos, para de esta forma poder transmitirlas por un "canal" o línea de comunicación, en forma de señal eléctrica, lumínicas, electromagnéticas, etc. Y de igual modo poder registrar o almacenar esta información, ya sea electrónica, magnética o mecánicamente, etc.

Definición: Si denominamos S = (s_1; s_2; .....; s_q) al conjunto de símbolos de un alfabeto dado; queda definida una codificación al poner en correspondencia todas las secuencias posibles de símbolos de S con secuencia de símbolos de algún otro alfabeto Z = (z_1; z_2; .....; z_r). Entonces "S" recibe el nombre de alfabeto fuente y "Z" alfabeto código.

4.2 CÓDIGO BLOQUE

Es aquel que asigna a cada uno de los símbolos del código fuente S una secuencia fija de símbolos del alfabeto código Z. Estas secuencias fijas (secuencias Z_j) reciben el nombre de palabras código.

S_1 Z_1 (z_{11}; z_{12}; .....; z_{1r})
S_2 Z_2 (z_{21}; z_{22}; .....; z_{2r})

S_j Z_j (z_{j1}; z_{j2}; .....; z_{jr})

4.3 CÓDIGO NO-SINGULAR

Es aquel en que todas sus palabras son distintas:

S_1 1
S_2 11
S_3 00
S_4 10

4.4 CÓDIGOS UNÍVOCAMENTE DECODIFICABLES

Se dice que un código es unívocamente decodificable si a cualquier secuencia de palabras código, le corresponde una y sólo una secuencia de símbolos de la fuente.

Por Ejemplo

S_1 1
S_2 11
S_3 00
S_4 10

Si se recepciona: 1100 *puede corresponder a* S_2 *y* S_3 *o a* S_1, S_1 *y* S_3.
Veamos que este código es no singular al considerar los símbolos individuales, pero es singular considerando alguna secuencia.

Por lo que podemos dar otra definición de código unívoco: Un código bloque se dice unívocamente decodificado sí y solo sí su extensión de orden n es no singular para cualquier valor finito de n.

Consideremos una codificación cuyas palabras código son de igual longitud. En este caso, indudablemente el código es unívoco. (siempre que sea no singular el original).

Otro caso son los llamado códigos "coma"; este nombre se debe a que el comienzo o el fin de cada palabra está señalizada por un cero o por un uno.

Ejemplo

	Código A	Código B
S_1	0	1
S_2	01	01
S_3	011	001
S_4	0111	0001

De esta forma se identifica cada palabra.

4.5 Códigos Instantáneos

Retomando el ejemplo anterior, donde los códigos A y B eran unívocos. Si estamos recepcionando palabras según el código A no seriamos capaces de decodificarlas instantáneamente según las vamos recibiendo. Por ejemplo, si recibimos la palabra 01, no podemos decir que corresponde a S_2 hasta que hayamos recibido el símbolo siguiente. Si el próximo es un 0 (cero) entonces si decimos que correspondía a S_2, si es un 1 (uno) debemos esperar el próximo, etc.

Si en cambio, utilizamos el código B, podemos decodificar las palabras instantáneamente.

CONDICION **I:** Sea $Z_i = z_{i1}; z_{i2}; .....; z_{im}$ una palabra de código. Se denomina prefijo de esta palabra a la secuencia $z_{i1}; z_{i2}; .....; z_{ij}$ con $j \leq m$.

A partir de esto se puede enunciar: "La condición necesaria y suficiente para que un código sea instantáneo es que ninguna palabra del código coincida con el prefijo de otra.

4.6 Inecuación de Kraft

Teorema

Si los enteros $l_1; l_2; .....; l_q$ satisfacen la inecuación:

$$\sum_{i=1}^{q} r^{-l_i} \leq 1 \qquad (4.6.1)$$

Entonces esta es condición suficiente y necesaria para que exista al menos un código instantáneo de r símbolos en su alfabeto y cuyas q palabras códigos tienen las longitudes l_1; l_2; .....; l_q respectivamente.

Nota: el teorema no dice que cualquier código cuyas longitudes cumplan la condición (4.6.1) es un código instantáneo.

Ejemplo

S_1 _____ 0

S_2 _____ 00

S_3 _____ 11

$$\sum_{i=1}^{3} 2^{-l_i} = 2^{-1} + 2^{-2} = 1/2 + 1/4 + 1/4 = 1$$

este código satisface la ec.(4.6.1) sin embargo no es un código, instantáneo. El teorema dice que existe algún código de longitud 1,2,2 que es instantáneo, por ejemplo el:

S_1 _____ 0

S_2 _____ 10

S_3 _____ 11

4.7 Ecuación de Mc Millan

Dado que los códigos instantáneos son una sub-clase de los unívocos, la condición suficiente para que exista, al menos un código unívoco, es que las longitudes l_i verifiquen la expresión:

$$\sum_{i=1}^{q} r^{-l_i} \leq 1 \qquad (4.7.1)$$

es decir, si las longitudes l_1; l_2; .....; l_q satisfacen la relación pueden construirse con ellas un código unívoco. La condición necesaria fue demostrada por Mc. Millan.

4.8 LONGITUD MEDIA DE UN CÓDIGO

Hasta ahora hemos visto las condiciones que debe cumplí un código instantáneo. Dado el caso real de una fuente de información que emite símbolos. Codificamos con A y B.

	Código A	Código B
S_1	00	1
S_2	10	10
S_3	11	110

Existiendo la probabilidad de crearse otros distintos de los precedentes. Entonces, ¿cuál es el más conveniente desde el punto de vista de la economía?

Indudablemente, cabría la pregunta: ¿Qué frecuencia de aparición tienen los símbolos?

DEFINICIONES:

Frecuencia Relativa: la frecuencia relativa de aparición de un evento es la suma de las veces que este aparece dividido por la cantidad total de eventos que componen la muestra.

Ejemplo:

→ a b c a a c b b a b a a c c a a a b -es una muestra-

frecuencia relativa de a = $^{9}/_{18} = {}^{1}/_{2}$

Muestra Significativa o representativa: es aquella en que podemos tomar a la frecuencia relativa como probabilidad.

Pasamos a codificar a los eventos “a”, “b” y “c”.

a - - - - con palabra código de longitud l_a

b - - - - con palabra código de longitud l_b

c - - - - con palabra código de longitud l_c

Entonces la longitud de la muestra (4.8.1) con los eventos codificados será

$$L_M = l_a \cdot N_a + l_b \cdot N_b + l_c \cdot N_c$$

donde N_i es el número de veces que aparece el evento i en la muestra.

$$L \text{ (longitud promedio)} = {}^{LM}/_{TM} = \underbrace{{}^{N_a}/_{TM}}_{f_a} l_a + \underbrace{{}^{N_b}/_{TM}}_{f_b} l_b + \underbrace{{}^{N_c}/_{TM}}_{f_c} l_c$$

donde:

TM: total de eventos de la muestra.

f_i: es la frecuencia relativa del suceso i.

Si la frecuencia es representativa, la frecuencia relativa coincide con la probabilidad del suceso a.

Luego:

$$L = P(a)\, l_a + P(b)\, l_b + P(c)\, l_c$$

RESUMIENDO:

Dado un código bloque que asocia los símbolos de una fuente S_1; S_2; .....; S_N con las palabras Z_1; Z_2; .....; Z_N y las probabilidades de los símbolos son P_1; P_2; .....; P_N. La **longitud media** del código L se define como:

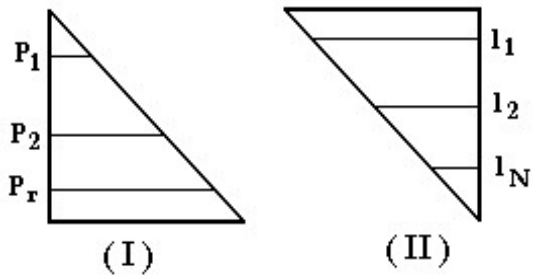
$$L = \sum_{i=1}^{N} P_i \; l_i$$

Desde el punto de vista económico nos interesan aquellos códigos en que la longitud es mínima.

Definición: dada la fuente S diremos que un código instantáneo aplicado a S es compacto (respecto a S) si su longitud media es igual o menor que la longitud media de todos los códigos instantáneos que puedan aplicarse a la misma fuente y al mismo alfabeto.

Es indudable que para obtener una longitud mínima, si el esquema de probabilidades tiene la forma (I) el esquema de longitudes deberá ser de la forma (II).

La mayor probabilidad – menor longitud de palabra



Si la fuente tiene un esquema equiprobable de símbolos, las longitudes de las palabras códigos se independizan del esquema de probabilidades.

Dado que las P_i son ≥ 0 y las $l_i \geq 0$ de (4.8.1) resulta que $L \geq 0$.

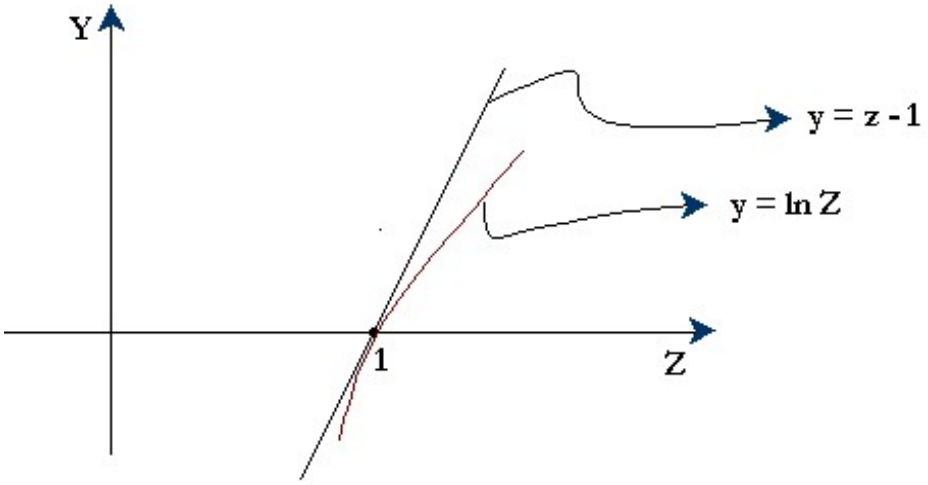
4.9 COTA INFERIOR DE LA LONGITUD MEDIA.

Dado que a nosotros nos interesa obtener una longitud mínima de la codificación, comenzaremos con una pregunta; ¿Hasta dónde podemos disminuir la longitud media?

Teorema: La longitud media de un código es mayor o igual que la entropía de la fuente dividida por el $\log_2$ del número de símbolos del alfabeto código.

$H(s)/\log r \leq L$ (logaritmo base 2 de x: log x)

Demostración:



Es fácil demostrar que: $\ln x^{z-1}$ **(4.9.1)**
multiplicando (4.9.1) por –1 :

$$\ln {}^{1}/_{X} \geq 1\text{-}z \qquad \textbf{(4.9.2)}$$

Sean Z (z_1; z_2; .....; z_q) e Y(y_1; y_2; .....; y_q) dos conjuntos de probabilidades; es decir: $z_i \geq 0$, $y_j \geq 0$ $\forall i,j$

y $\sum_{i=1}^{q} z_i = \sum_{j=1}^{q} y_j = 1$ **(4.9.3)**

podemos escribir, haciendo uso de: $\log_a z = {}^{1}/\log_b a$

$$\sum_{i=1}^{q} z_i \log y_i/z_i = \sum_{i=1}^{q} z_i (\ln 2)^{-1} \ln y_i/z_i$$

o sea:

$$\sum_{i=1}^{q} z_i \log y_i/z_i = \overbrace{1/\ln 2 \sum_{i=1}^{q} z_i \ln y_i/z_i}^{I}$$

Y aplicando la inecuación (4.9.1) a cada término de la $\sum$ (I) obtenemos:

$$\sum_{i=1}^{q} z_i \log y_i/z_i = 1/\ln 2 \sum_{i=1}^{q} z_i \ln y_i/z_i \leq 1/\ln 2 \sum_{i=1}^{q} z_i (y_i/z_i - 1) =$$
$$= 1/\ln 2 \sum_{i=1}^{q} z_i ((y_i/z_i)/z_i) = 1/\ln 2 (\sum_{i=1}^{q} y_i - \sum_{i=1}^{q} z_i) = 0$$

Luego:

$$\Longrightarrow \sum_{i=1}^{q} x_i \log 1/x_i \leq \sum_{i=1}^{q} x_i \log 1/y_i \qquad (4.9.5)$$

$$\sum_{i=1}^{q} x_i \log y_i \leq \sum_{i=1}^{q} x_i \log x_i \Longrightarrow$$

$$\sum_{i=1}^{q} x_i \log y_i/x_i \leq 0 \quad \text{lo que significa:}$$

que será una igualdad, ∀ valor de i si y sólo si $z_i = y_i$

Consideremos una fuente de memoria de símbolos S_1; S_2; .....; S_q con sus probabilidades P_1; P_2; .....; P_q. Supongamos un código bloque que codifica a la fuente S_1 con un alfabeto de r símbolos y l_i la longitud de la palara que codifica a S_i. La entropía valdrá:

$$H(s) = -\sum_{i=1}^{q} P_i \log P_i$$

Supongamos además que Q_1; Q_2; .....; Q_q son números tales que $Q_1 \geq 0$

$$\forall_i \quad y \quad \sum_{i=1}^{q} Q_i = 1$$

Luego, de acuerdo a la (4.9.5) podemos escribir:

$$\sum_{i=1}^{q} P_i \log 1/P_i = H(s) \leq -\sum_{i=1}^{q} P_i \log Q_i \qquad (4.9.6)$$

Igualdad para todo i si y sólo $Q_i = P_i$

La (4.9.6) es valida para conjunto de números positivos, Q_i, cuya suma sea la unidad. En particular para:

$$Q_i = r^{-l_i} / \sum_{i=1}^{q} r^{-l_i}$$

Luego:

$$H(s) \leq -\sum_{i=1}^{q} P_i \log (r^{-l_i}) + \sum_{i=1}^{q} P_i \log \sum_{j=1}^{q} (r^{-l_j}) \leq \log r \underbrace{\sum_{i=1}^{q} P_i \, l_i}_{I} + \log \underbrace{\sum_{j=1}^{q} (r^{-l_j})}_{II} \quad (4.9.6)bis$$

Si exigimos que el código sea instantáneo por Kraft; (II) es ≤ 1 ⇒ log. (II) ≤ 0

- **H(s) ≤ log r L –**
- **H(s) / log r ≤ L donde H(s) = bits**

Pudiendo expresar el cociente H(s) / log r, como la entropía en unidades r-arias quedando:

Hr(s) ≤ L **(4.9.7)**

O sea la (4.9.7) expresa que nunca podremos obtener un código cuya longitud media sea menor que la Hr(s) (entropía en unidades r-arias).

Por otro lado, hemos puesto en relación dos magnitudes: la Hr(s) con otra, la longitud media, que no depende de H(s).

4.11 FUENTES PARTICULARES. CODIFICACIÓN

4.12 PRIMER TEOREMA DE SHANNON

4.13 Método de Huffman para la construcción de Códigos Compactos

4.13.1 Código Compacto Binario

Veremos un método para crear un código compacto en el caso de un alfabeto binario, para luego generalizar para un alfabeto r-ario.

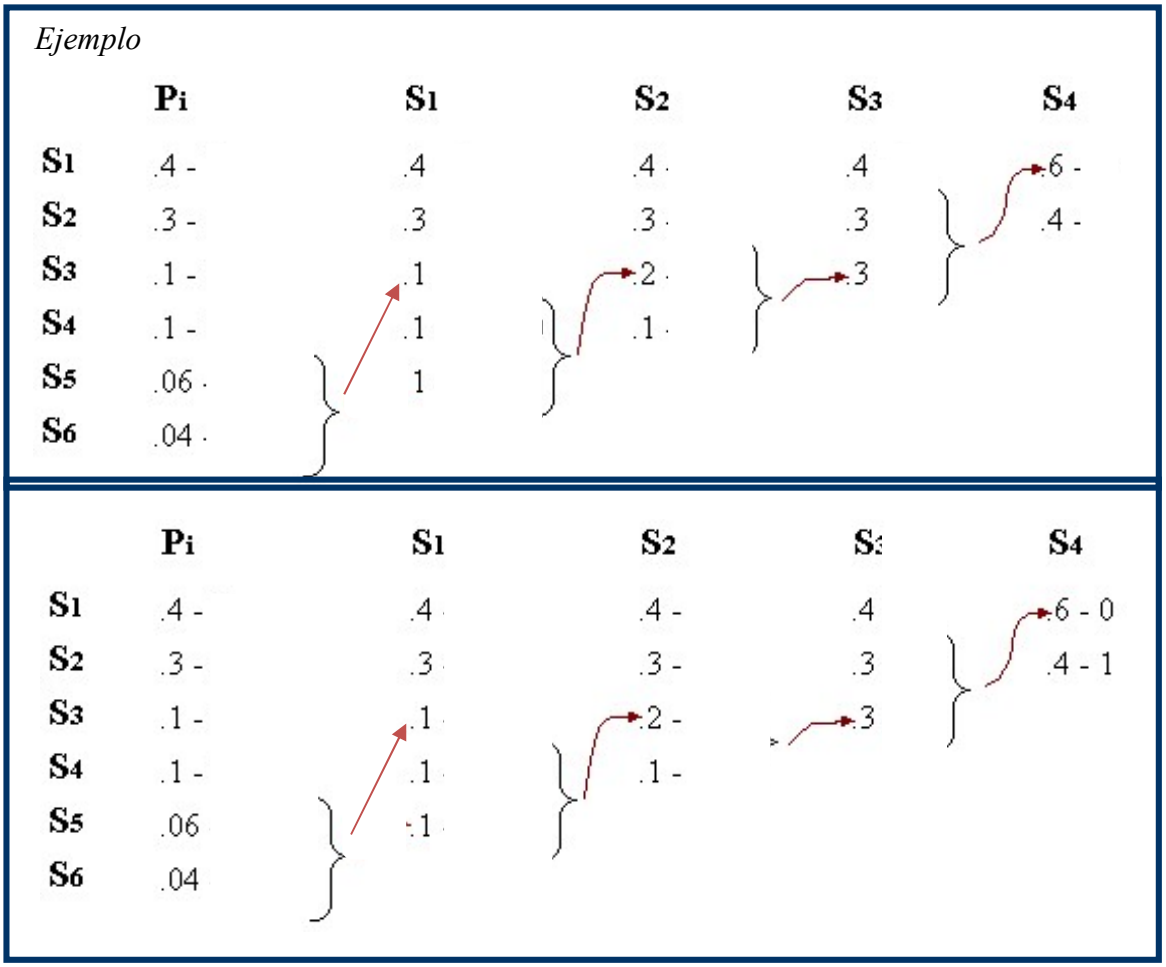
Ambos procedimientos fueron desarrollados por huffman.

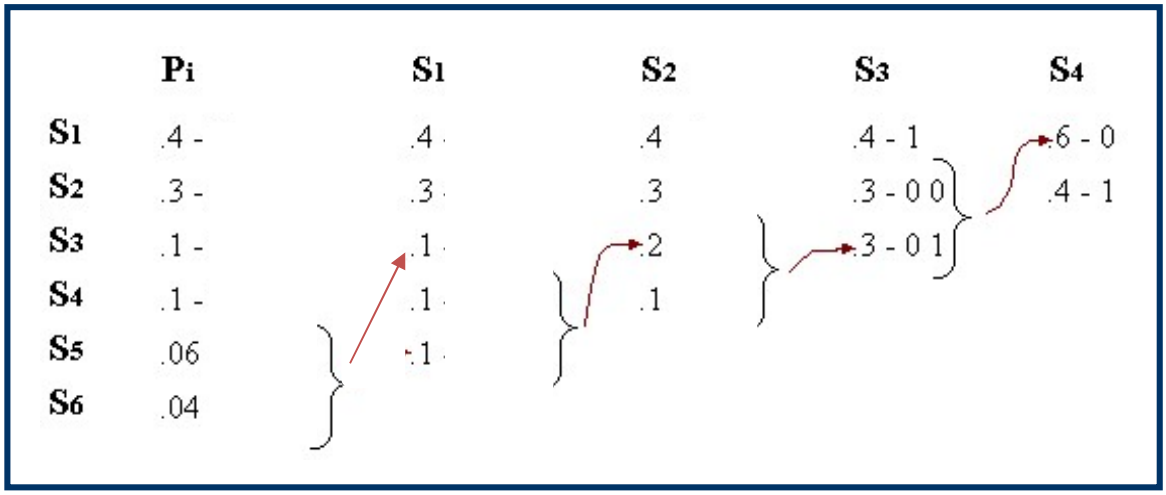
Consideremos una fuente S de símbolos $S_1, S_2, \ldots\ldots, S_q$ y probabilidades $P_1; P_2; \ldots\ldots; P_q$. Supongamos ordenarlos de forma que: $P_1 \geq P_2 \geq \ldots\ldots \geq P_q$.

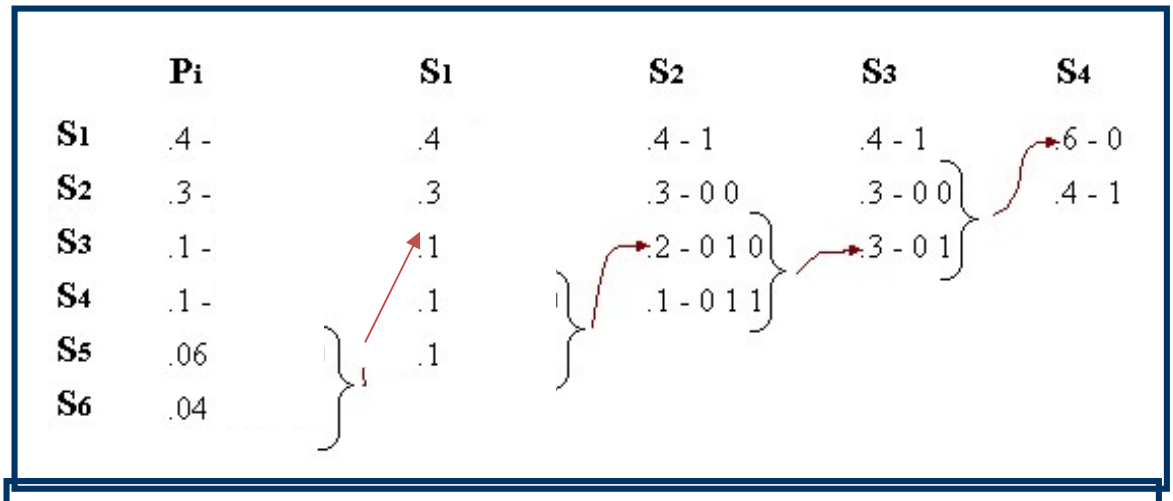
Si los dos últimos símbolos los fundimos en uno solo (sumando sus probabilidades), se tiene una nueva fuente de q-1 símbolos. La denominaremos fuente reducida de S. Los nuevos símbolos los ordenamos de igual forma que anteriormente, y realizamos el agrupamiento de los dos últimos. Así sucesivamente hasta llegar a una fuente de dos símbolos. Este es el primer paso.

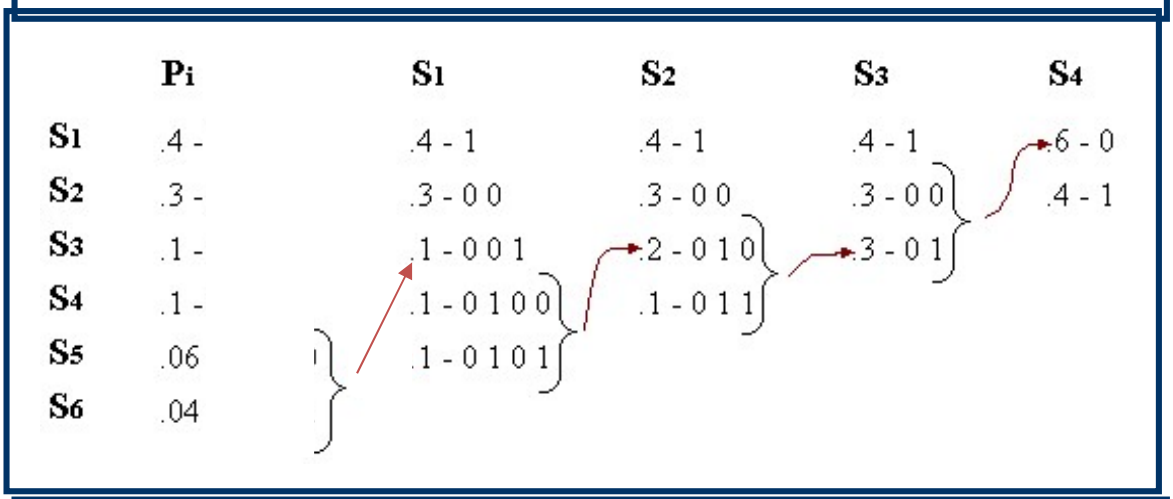
El segundo paso consiste en asignarle un código compacto a la última fuente reducida, que precisamente es 0 y 1. El paso siguiente es retroceder en el procedimiento hasta que, habiendo partido de un código compacto, se obtenga en S_1 un código compacto.

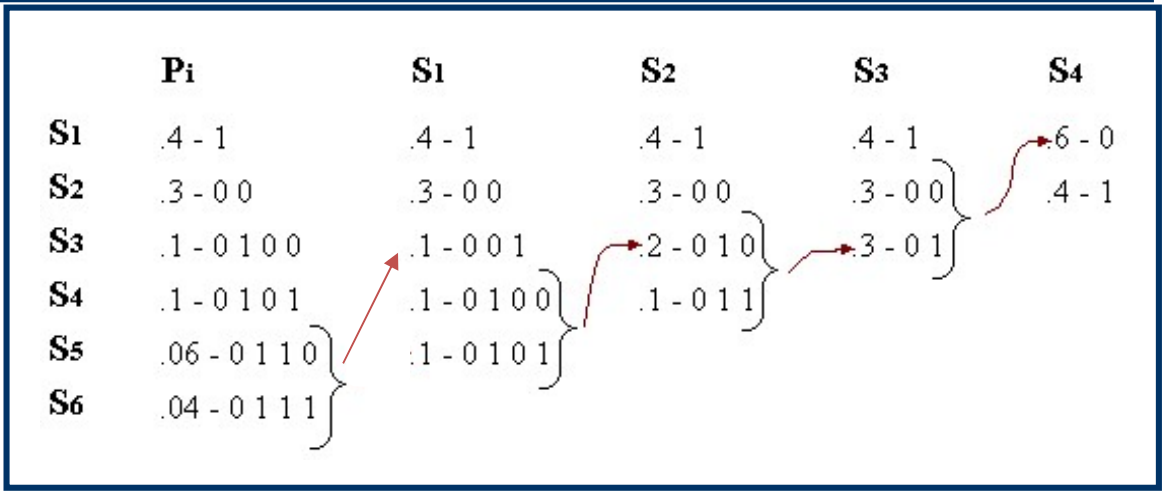
Ejemplo




Pi S1 S2 S3 S4
S1 .4 - .4 .4 .4 - 1 6 - 0
S2 .3 - .3 .3 .3 - 0 0 .4 - 1
S3 .1 - .1 .2 3 - 0 1
S4 .1 - .1 .1
S5 .06 .1
S6 .04


Pi S1 S2 S3 S4
S1 .4 - .4 .4 - 1 .4 - 1 6 - 0
S2 .3 - .3 .3 - 0 0 .3 - 0 0 .4 - 1
S3 .1 - .1 .2 - 0 1 0 3 - 0 1
S4 .1 - .1 .1 - 0 1 1
S5 .06 .1
S6 .04


Pi S1 S2 S3 S4
S1 .4 - .4 - 1 .4 - 1 .4 - 1 6 - 0
S2 .3 - .3 - 0 0 .3 - 0 0 .3 - 0 0 .4 - 1
S3 .1 - .1 - 0 0 1 .2 - 0 1 0 3 - 0 1
S4 .1 - .1 - 0 1 0 0 .1 - 0 1 1
S5 .06 .1 - 0 1 0 1
S6 .04


Pi S1 S2 S3 S4
S1 .4 - 1 .4 - 1 .4 - 1 .4 - 1 6 - 0
S2 .3 - 0 0 .3 - 0 0 .3 - 0 0 .3 - 0 0 .4 - 1
S3 .1 - 0 1 0 0 .1 - 0 0 1 .2 - 0 1 0 3 - 0 1
S4 .1 - 0 1 0 1 .1 - 0 1 0 0 .1 - 0 1 1
S5 .06 - 0 1 1 0 .1 - 0 1 0 1
S6 .04 - 0 1 1 1

Al decir retroceder nos referimos a:

Sea S_j el código compacto correspondiente a una de las fuentes reducidas, uno de sus símbolos, S_α, estará formado por dos símbolos de la fuente procedente S_{j-1} que denominamos $S_{\alpha 0}$ y $S_{\alpha 1}$; todos los demás símbolos se identifican con uno solo de S_{j-1}.

Según esto el código compacto (instantáneo) correspondiente a S_{j-1} se deduce del S_j de acuerdo a la siguiente regla:

Se asigna a cada símbolo de S_{j-1} (excepto $S_{\alpha 0}$ y $S_{\alpha 1}$) la palabra correspondiente al símbolo de S_j. Las palabras correspondiente a $S_{\alpha 0}$ y $S_{\alpha 1}$ se forman añadiendo un 0 y un 1 respectivamente, a la palabra asignada a S_α.

Se pone en evidencia que en algunas ocasiones resulta innecesario continuar la secuencia de reducciones de la fuente original hasta la fuente de dos símbolos. Unicamente deberá reducirse hasta llegar hasta una reducción para la cual se pueda encontrar fácilmente un código compacto. Por ejemplo, si el esquema de probabilidades de los símbolos resulta $(½)^{\alpha i}$. A partir de aquí se vuelve hacia atrás de forma vista.

¿Este código hasta aquí es compacto?, demostramos que si.

Supongamos la reducción S_j.

$$\mathbf{L_j = P_{j1}\, l_{j1} + ... + P_{jn}\, l_{jn}}$$

Supongamos además que esta reducción está codificada de forma compacta con C_j.

La reducción anterior S_{j-1} tendrá una L_{j-1}.

$$\mathbf{L_{j-1} = P_{(j-1,\,1)}\ l_{(j-1,\,1)} + ...+ P_{(j-1,\,n)}\ l_{(j-1,\,n)} + P_{(j-1,\,n+1)}\ l_{(j-1,\,n+1)}}$$

Donde:

$$\mathbf{P_{(j-1,\,1)} = P_{(j,1)}}$$
$$\mathbf{P_{(j-1,\,1)} = l_{(j-1,\,1)}}$$

Además:

$$\mathbf{l_{(j-1,\,n)} = l_{(j-1,\,n+1)} = l_{(j,\,n)} + 1}$$
$$\mathbf{P_{(j-1,\,n)} = P_{(j-1,\,n+1)} = P_{(j,\,n)}}$$

Luego:

$$\mathbf{L_{j-1} = P_{j1}\ l_{j1} + ... + \{(\,l_{j,1} +1)\ (\,P_{(j-1,\,n)}\ P_{j-1,\,n} +1)\} =}$$
$$\mathbf{= P_{j1}\ l_{j1} + ... + l_{jn}\, P_{jn} + P_{(j-1,\,n)} + P_{(j-1,\,n+1)}}$$
$$\mathbf{L_j \qquad\qquad P_{\alpha 0} \quad P_{\alpha 1}}$$

O sea:

$$\mathbf{L_{j-1} = L_j + P_{\alpha 0} + P_{\alpha 1}} \qquad \mathbf{(4.13.1.1)}$$

Donde $P_{\alpha 0}$ y $P_{\alpha 1}$ son las probabilidades de los símbolos $S_{\alpha 0}$ y $S_{\alpha 1}$: símbolos que provienen de haber desdoblado el S_α; donde S_α es el símbolo menos probable de la reducción compacta S_j.

DEMOSTRACIÓN

Se desea demostrar que si C_j es compacta, C_{j-1} también lo es.

Se hará por reducción al absurdo, queremos demostrar que si L_j la menor longitud media de un código instantáneo correspondiente a S_j; L_{j-1} (dado por la ecuación 4.13.1.1) es también la longitud media mínima para S_{j-1} de longitud media L_{j-1} y sean $X'_1, X'_2, X'_{\alpha 0}, X'_{\alpha 1}$, las palabras de dicho código de longitudes l'_{j-1}, ..., $l'_{\alpha 0}$, $l'_{\alpha 1}$.

Supongamos los subíndices ordenados según el orden decrecientes de las probabilidades de los símbolos respectivos, es decir

$$\mathbf{l'_1 < ..... < l_{\alpha 0} < l_{\alpha 1}}$$

Ahora construyamos C_j, código correspondiente a S_j, combinando $X'_{\alpha 0}$ y $X'_{\alpha 1}$ y suprimiendo el último binit sin alterar los demás. El resultado es un código instantáneo para S_j; de longitud media L'_j; que debe cumplir.

$$\mathbf{L' = L'_j + P_{\alpha 0} + P_{\alpha 1}}$$

Esta ecuación comparada con la (4.13.1.1) implica que existe un código de longitud media $L'_j < L_j$ lo que es absurdo ya que se puso que el C_j era compacto.

4.13.2 CÓDIGO COMPACTO R-ARIO

Por extensión, se puede ver que la obtención de un código compacto r-arios es similar al binario. Salvo que los símbolos se agrupon de a "r".

De esta manera cada fuente reducida tendrá r-1 símbolos menos que la anterior. ¿Pero en la última reducción habrá r símbolos?

Esta pregunta es afirmativa si se cumple que, la fuente original esta formada por ω símbolos donde $\boldsymbol{\omega = r + \alpha(r-1)}$ siendo α un número entero.

Si esto no ocurre se deberá añadir símbolos "falsos", en número suficiente como para que se cumpla la (4.15.2.1). A los falsos símbolos se les asigna probabilidad nula, de modo que pueden ser ignorados una vez que el código haya sido construido.

4.14 RENDIMIENTO Y REDUNDANCIA DE UN CÓDIGO

Se define rendimiento de un código a:

$$R = \frac{Hr(s)}{L} \qquad (4.14.1)$$

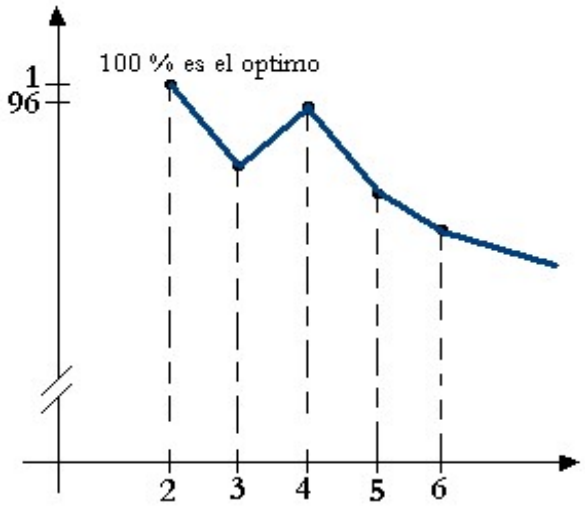
Como $H_r(s) < L$ implica ≤ 1

Así mismo puede definirse la redundancia de un código según la expresión:

$$\Re = 1 - n = \frac{L - Hr(s)}{L}$$

Es interesante graficar el n en función del n° de símbolos del código.

Si el alfabeto código tiene r símbolos, y el alfabeto de la fuente tiene un esquema de probabilidades $(¼)^{xi}$; resulta:



4.15 CONFIABILIDAD (VS. ECONOMÍA)

La información y codificación de una fuente, cuando se transmite la información por un canal real, el ruido de este cambia la información que se desea transmitir.
O sea:

Se emite 001 se recepciona 011

Es imprescindible entonces, estudiar la forma de poder detectar u aún más corregir ciertos tipos de errores introducidos en el canal.

Se define **Ruido Unitario** al cambio, en una palabra de un binit por su complemento (en codificación binaria).

4.15.1 PARIDAD

Este es un criterio de detección de error que consiste en agregar a la palabra codificada un bimit (ip) más (redundante), de forma tal que éste tome el valor.

(a) $\mathbf{ip = i_1 + i_2 + ... + i_n}$ **(4.15.1.1)**

si se adapta el criterio de paridad par y

(b) $\mathbf{ip = i_1 + i_2 + ... + i_n + 1}$ **(4.15.1.2)**

donde: i_i, con i = 1 .....n, son los binit que conforman la palabra código.

ip : binit de paridad (impar o par).
⊕ : función "or exclusive".
n : longitud de la palabra código.

Luego la palabra que se transmite será:

$\mathbf{i_1, i_2, ..., i_n, ip}$

Al recepcionar se controla si ip cumple con la (4.15.1.1) si se utiliza paridad par o con la (4.15.1.2) si se utiliza paridad impar.

Si en el camino se produjo algún error unitario se altera la paridad de la palabra, siendo posible detectarlo en el destino.

Es evidente que también se detectan los cambios de orden K toda vez que K sea impar.

Con este tipo de control no solo se puede llegar a detectar sino también a corregir de la siguiente forma:

Antes de transmitir se forman grupos de palabras, de forma tal que conformando una matriz de m x n; se completan las filas y las columnas con el binit de paridad de la fila y de la columna obteniéndose una nueva matriz (m +1) x (n +1).

De esta forma, si al recepcionar esta matriz; el canal ha introducido un ruido unitario y se alteró, por ejemplo i_{22}, al controlar paridad, el recepcionista verá que no cumplen la regla i_{2f} r i_{2c}, quedando unívocamente identificado el binit alterado.

Si el alterado es una de los binit de paridad, es detectado por el incumplimiento de la paridad i_{pt}.

¿ Dónde se pone en evidencia la redundancia, como factor de absorción del ruido?

Pensemos: tenemos 3 símbolos en nuestro alfabeto código. Con esto podemos codificar 2^3 símbolos de una fuente, o sea 8 símbolos. Si ahora agregamos un binit de paridad seguiremos codificando 2^3 símbolos (ahora con protección).

Pero con 4 símbolos en el alfabeto código, se pueden codificar 2^4 símbolos fuente o sea 16.

De esto se deduce que no se utilizan:

$$2^4 - 2^3 = 3 \text{ configuraciones}$$

Estas son precisamente las configuraciones que nos absorben el ruido.

4.15.2 Por Mayoría

Este método consiste en codificar de forma tal que una palabra haya mayor "peso" de "1s" o de "0s" de forma tal que al recepcionar se controla so se violó esa regla.

Ejemplo:

Codificaremos con una longitud constante, l = 5 y peso 3 de unos.

Existen un número de palabras que cumplen con este peso igual a:

$$C_5^3 = \frac{V_5^3}{i_3} = \frac{5 \times 4 \times 3}{3 \times 2 \times 1} = 10$$

0 0 1 1 1
0 1 1 1 0
1 1 1 0 0
1 0 1 0 1
1 1 0 0 1
0 1 0 1 1
1 1 0 1 0
1 0 0 1 1
1 0 1 1 0
0 1 1 0 1

Nótese que con cinco símbolos en un alfabeto código, se pueden codificar (sin protección) $2^5 = 32$ símbolos fuentes. Este método detecta todos los errores unitarios.

En los errores de orden mayor no se puede asegurar su eficacia.

4.15.3 DE IGUAL PESO

Es muy similar al de mayoría, salvo que ahora se codifica de forma tal de que el número de "1s" sea igual al número de "0s".

Indudablemente, las palabras códigos resultarán de longitud l, con l par.

4.16 REGLA DE DECISIÓN

Hemos visto hasta ahora, que algunos criterios de protección detectan errores pero no los corrigen.

En algunas aplicaciones, esto se soluciona pidiendo (en el caso de haber detectado un error) que se retransmita la información.

En otras aplicaciones, esto no es posible. En estas aplicaciones entonces, se debe buscar otra solución.

Esta puede ser adoptar una *Regla de Decisión*.

Consiste en recibida una palabra que se sabe con equivocación: modificarla según una regla, de manera tal que adopte la forma de una palabra sin equivocación.

Es indudable que esto trae asociado una improbabilidad de error (o acierto).

De ahí que esta regla de decisión debe ser adoptada de forma tal que proporcione una probabilidad de error, lo más baja posible.

Ejemplo:

Símbolo Fuente	**Palabra Código**
S_1	**000**
S_2	**111**

Si sabemos que el canal tiene una probabilidad relativamente elevado debe provocar cambios unitarios y en mucho menor grado, errores de orden superior.

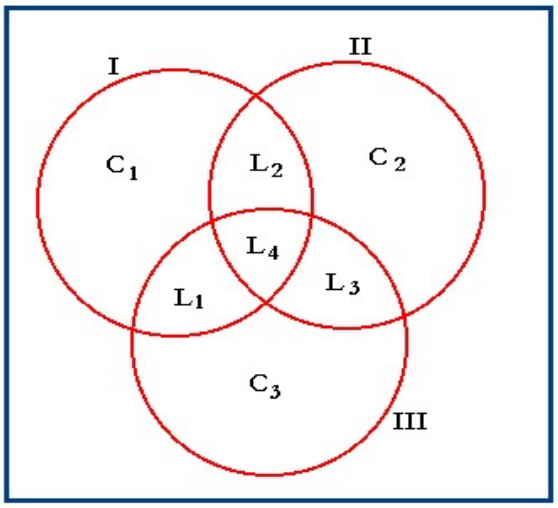
La regla de decisión es evidente:

Si se recibe: 001 se decodifica como recibido 000
011 se decodifica como recibido 111
.....
.....

4.17 HAMMING (PARIDAD POR ZONAS)

Este es un método que consiste en controlar la paridad de zonas. Estas zonas son forradas por conjuntos de binits.

Veamos en un diagrama de Venn:



Acá los binits que transportan información (y por lo tanto juegan libremente) son i_0, i_1, y i_3.

Pero se observa que las zonas delimitadas, poseen además otros binits, C_1, C_2 y C_3 que son los binits de control.

Se los denomina binits de control, pues estos se acomodan de forma tal que, conjuntamente con los binits de información de su zona, conserven una determinada paridad.

$$\mathbf{C_1 = i_0 \oplus i_1 \oplus i_2}$$
$$\mathbf{C_2 = i_0 \oplus i_2 \oplus i_3}$$
$$\mathbf{C_3 = i_0 \oplus i_1 \oplus i_3}$$

Recordar que, C_1,C_2, C_3, i_0, i_1, i_2, i_3 pueden adoptar los valores 0 ó 1.

Luego:

C_1 controla a i_0, i_1, i_2 y a C_1 **(4.17.4)**

C_2 controla a i_0, i_2, i_3 y a C_2 **(4.17.5)**

C_3 controla a i_0, i_1, i_3 y a C_3 **(4.17.6)**

La palabra a ser transmitida ahora tendrá la forma:

$$\mathbf{i_0\ i_1\ i_2\ i_3\ C_1\ C_2\ C_3} \qquad \textbf{(4.17.7)}$$

Al recepcionar se controla la paridad de las zonas.

Por ejemplo: Supongamos que se alteró el valor de i_2.

Esto trae aparejado que las ecuaciones que no se cumplen son (4.17.1) y la (4.17.2), o en otras palabras se modifico la paridad de las zonas I y II quedando unívocamente determinado el binit cambiado: i_2.

Si el error proviene de un cambio en un binits de control; esto se pone en evidencia pues la zona modificada es solo una, la correspondiente a dicho binit de control.

PREGUNTA: si se realiza un código con m dígitos de información. ¿Cuántos dígitos de control se deben agregar?

Se deben agregar n dígitos de control.

Con n dígitos de control se puede tener 2^n configuraciones distintas.

Cada una de estas configuraciones, indicará error en un dígito, existiendo m + n dígitos. Pero además deberá existir una configuración que indique la no existencia de error. Luego se deberá cumplir con la relación:

$$2^n \geq m + n + 1 \qquad (4.17.8)$$

En nuestro ejemplo, m = 4, luego m debe valer 3 para que:

$$23 \geq 4 + 3 + 1$$ se verifique

Hagamos una parte del problema

	digitos de informacion				digitos de control			
	i_0	i_1	i_2	i_3	C_1	C_2	C_3	
	0	0	0	1	0	1	1	
	0	0	1	0	1	1	0	
	0	0	1	1	1	0	1	*
16	1	1	0	0	1	0	1	
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	
	1	1	1	1				

Notar que las configuraciones de los dígitos de control se repiten pero existen ocho configuraciones (que no son las mostradas en *) que me indican la:

(a)	equivocación en dígitos de información		**4**
	equivocación en dígitos de control.	**3**	
(b)	La configuración que dice todo O.K.	**1**	
	Total	**8**	

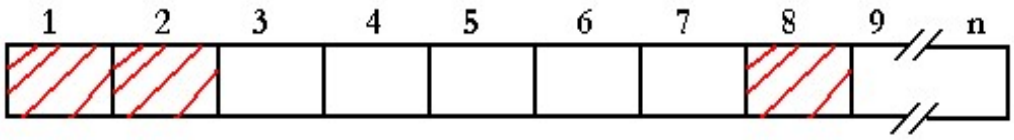
Vimos que, chequeando las zonas cuyas paridades se han alterado según sea utilizando el diagrama de Venn, o las tablas (16.5.4) (16.5.5), (16.5.6) se identifica el dígito que proporciona el problema. Este es el método cuando la palabra protegida se envía de la forma mostrada en (16.5.7).

Pero existe otro método que, indica como elegir las zonas; y como intercalar, entre los dígitos de información, los dígitos de control; de forma tal que (enumerando la posición de los dígitos en la palabra transmitida), si se produce un error, la configuración que adoptan los dígitos del control, debido a ese error, me indica (en decimal) el n° del dígito que fue alterado. Falta aclarar que la configuración de los dígitos de control se obtiene así.

Si la Cj fue alterada, el dígito de control correspondiente se coloca en 1. De lo contrario en cero. O sea zona C_j Alterada $C_j = 1$

zona C_j no Alterada $C_j = 0$

Veamos ahora como numeramos e intercalamos los dígitos de control en la palabra codificada.



Esta es la palabra y se han numerado los dígitos. Los dígitos de control se intercalan en las posiciones 2^n con n ∃ IR (dígitos rayados).

Los dígitos que quedan libres, se usan para transportar información.

Falta ahora ver como elegimos las zonas.

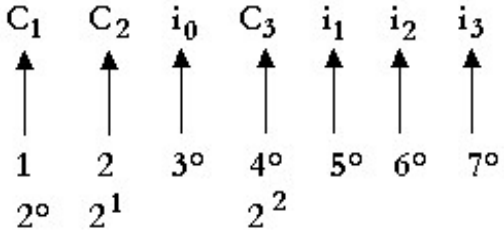
Ya que la elección de esta depende la identificación del dígito alterado, esta elección no puede ser arbitraria.

Ejemplo:

3 binits de control

4 binits de información

En la palabra las posiciones serán las siguientes:



Recordemos ahora lo que deseamos obtener:

(a) Se coloca un "uno" en el dígito de control si al recepcionar, se detecta que en esa zona no se conserva la paridad pre-establecida.

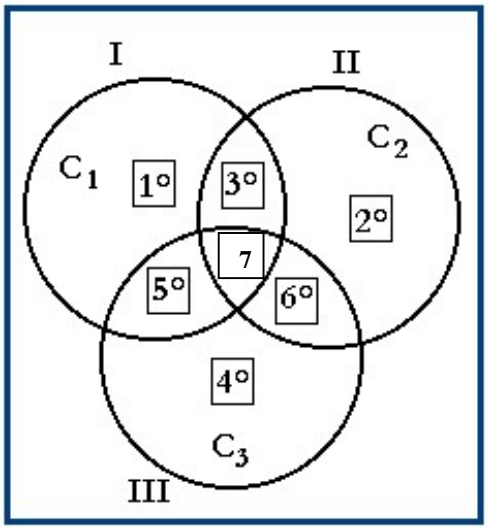
(b) Colocando luego los dígitos de control en orden; el n° formado en decimal, es el número del dígito alterado.

Ejemplo:

Zona I alterada ⟶ **$C_1 = 1$**
Zona II alterada ⟶ **$C_2 = 1$**
Zona III correcta ⟶ **$C_3 = 0$**

C_1	**C_3**	**C_2**			
0	0	1	⟶	3	⇒ alterado el 3° dígito

Diagrama propuesto:



Luego:

C_1 { 1, 3, 5, 7 }
C_2 { 2, 3, 6, 7 } = (16.5.9)
C_3 { 4, 5, 6, 7 }

Si lo planteamos sobre un diagrama de Karnauch, este debe tener 8 cuadraditos.

C_3 \ C_2 C_1	00	01	11	10
0	O.K.	C_1 1°	3°	C_2 2°
1	C_3 4°	5°	7°	6°

Cada cuadrito, representa, en decimal, el n° del dígito. El 000, lo reservamos para indicar que no hubo ningún error; pues esto, implica $C_1 = 0$, $C_2 = 0$, $C_3 = 0$.

La reglas prácticas para determinar las zonas que controlan C_1, C_2, C_3 es:

(a) Se ubican los dígitos de control en sus respectivos lugares.

(b) Se observa que a cada dígito de control, por estar en las posiciones 2^n; le corresponde un n° en binario con un solo "1".

(c) Cada dígito de control, controla aquellos dígitos que en binario, posean un "1" en el binit en el cual dicho dígito de control lo posea.

Ejemplo:

C_2 le corresponde 0 1 0

Luego C_2 controla aquellos dígitos que en binario posean un "1" en el binit señalado con una | o sea:

0 1 1 3er dígito (i_0)
1 1 1 7° dígito (i_3)

0 1 0 **2do dígito (es C_2)**
0 1 1 **3er dígito (i_0)**
1 1 0 **6° dígito (i_2)**

Supongamos querer transmitir la palabra:

1 1 0 1

La ubicación será:

1 0 1 0 1 0 1

Ahora deben colocar, con el valor adecuado los dígitos de control.

De acuerdo a (16.5.9)

$$C_1 = 1 \oplus 1 \oplus 1 = 1 \qquad (**)$$
$$C_2 = 1 \oplus 0 \oplus 1 = 0$$
$$C_3 = 1 \oplus 0 \oplus 1 = 0$$

Luego la palabra que se transmite es:

1 0 1 0 1 0 1

Supongamos que al recepcionar se obtiene:

1 0 1 0 1 1 1

Se controla la paridad de la zonas nuevamente realizando el cálculo (**) Observando que:

zona I	**no alterada**	**$C_1 = 0$**
zona II	**alterada**	**$C_2 = 1$**
zona III	**alterada**	**$C_3 = 1$**

1 1 0 **6** **6° dígito alterado**

En este ejemplo, analizándolo en detalle se ve que se puede desechar cualquier dígito cumpliéndose siempre que la probabilidad de error doble no detectado es siempre igual.