Electrónica Digital

Ingeniería Informática – FICH, UNL
Leonardo Giovanini

Códigos

Organización de la clase

En esta clase se estudiarán los siguientes temas:

- Definiciones y ejemplos;
- Propiedades;
- Decodificación
- Usos.

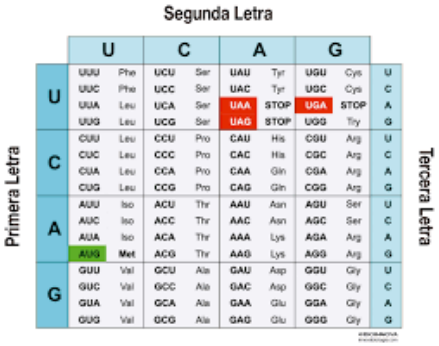
Códigos – *Definición*

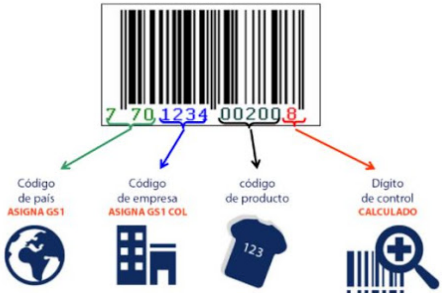
Un ***código*** es un ***sistema de reglas para convertir información*** (letras, palabras, sonidos, números, imágenes o gestos, etc) en otra forma o representación que tenga propiedades deseadas (legibilidad, lecturabilidad, portabilidad, tamaño o confidencialidad, etc.) para su transmisión, almacenamiento y procesamiento.

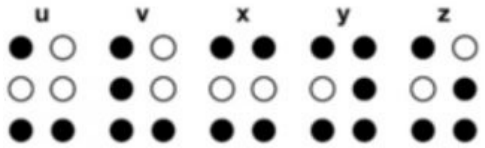
Un ejemplo temprano es el lenguaje que permitió a las personas comunicar lo que ven, escuchan, sienten o piensan.

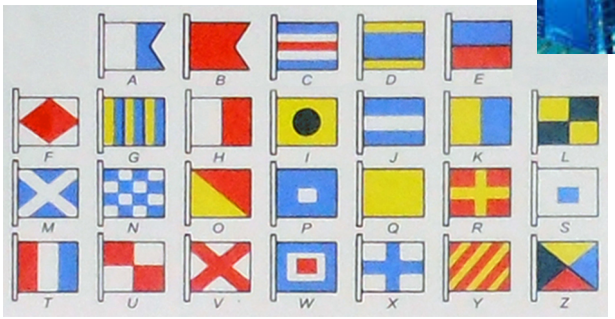
El habla limita el alcance de la comunicación a la audiencia presente cuando se pronuncia el discurso. La invención de la escritura convirtió el lenguaje hablado en símbolos visuales, extendiendo el alcance de la comunicación a través del espacio y el tiempo.

Utilizamos codigos para permitir la comunicación en lugares donde el lenguaje es difícil o imposible de utilizar. Por ejemplo, el semáforo, el código de banderas y el código Braile entre otros.



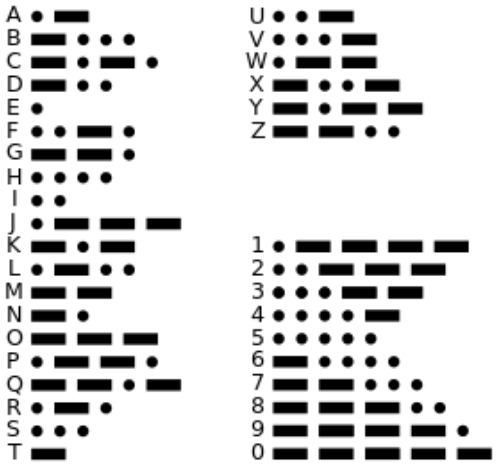




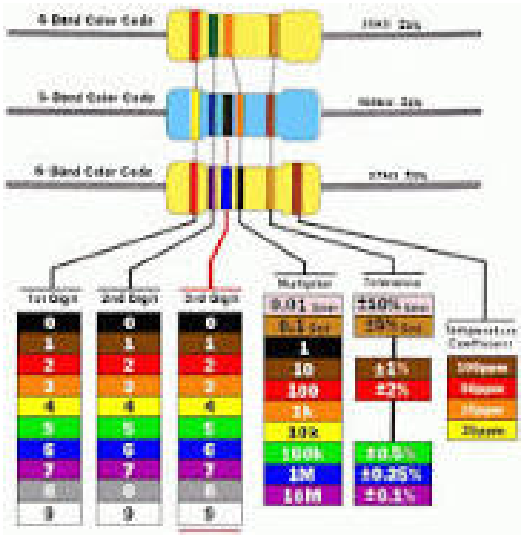


Códigos – *Ejemplos*

Ejemplos de códigos son



Código Morse se utiliza para transmitir mensajes;



Código de colores comunica los valores de resistencias y capacitores;

Señales de transito comunicar información a conductores.

Códigos – *Ejemplos*

Caracteres ASCII de control		
00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

Caracteres ASCII imprimibles					
32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido							
128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	▒	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	▒	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

Código ASCII se utiliza para almacenar, transmitir y procesar informacion en formato digital;

El conjunto de ***información original*** se conoce como ***conjunto fuente*** $\mathcal{S}$ (o alfabeto si el conjunto es finito) donde $s_i \in \mathcal{S} := \{s_1, \ldots, s_q\}$ aparece con una probabilidad $\mathbb{P}(\mathcal{S} = s_i)$.

El ***conjunto transformado*** se lo conoce como ***conjunto objetivo*** $\mathcal{T}$ (o alfabeto si el conjunto es finito) que está constituido por ***palabras códigos*** $t_i = C(s_i) \in \mathcal{T} := \{t_1, \ldots, t_n\}$.

Desde el punto de vista formal

Un código C es una función $C: \mathcal{S} \longrightarrow \mathcal{T}$ tal que para todo $s_i \in \mathcal{S}$ existe un $t_i = C(s_i) \in \mathcal{T}$.

El ***número de elementos*** de $\mathcal{T}$ utilizados por el código se llama ***tamaño del código***, tal que $\left| \mathcal{T} \right| \leqq n$.

Un código con q fuentes y tamaño $\left| \mathcal{T} \right| = M$ se lo denomina como un "código – (q, M)".

La ***extensión de un código*** es el mapeo de ***secuencias fuente*** a ***cadenas de palabras código*** de longitud finita, que se obtiene concatenando r símbolos

$$C(s_1, s_2, \ldots, s_r) = C(s_1)\, C(s_2) \ \ldots \ C(s_r),$$

de modo que el conjunto $\mathcal{T}$ de la extensión de un código tiene n^r simbolos.

La ***tasa de un código*** ΔC es una ***medida de su eficiencia***, formalmente se lo define como

$$\Delta C = n^{-1} \log_q M.$$

Se pueden especificar M palabras códigos usando símbolos $\log_q M$ cuando no hay redundancia. Cuanto más grande sea n, mayor será la cantidad de palabras código no utilizadas ($\log_q M$ son suficientes y por lo tanto hay $n - \log_q M$ símbolos adicionales que son redundantes).

Las propiedades que más se estudían de un código son:

- **Longitud** – dada la longitud de las palabras código $l_w(C(s_i))\ i=1,\dots,q$; que es la cantidad de símbolos elementos que las componen, de modo que la longitud de un código $l(C)$ se calcula como

$$l(C)=\sum_{s_i\in\mathcal{S}} l_w(C(s_i))\mathbb{P}(\mathcal{S}=s_i)$$

De acuerdo con su longitud, los códigos se clasifican en

- **Longitud fija** – es un código en el cual un número fijo de símbolos es codificado en un número de fijo de símbolos de salida;
- **Longitud variable** – es un código en el cual un número fijo de símbolos es codificado en un número de variable de símbolos de salida.

Ejemplo – Consideremos el ***alfabeto fuente*** $\mathcal{S}=\{a,b,c\}$ y el ***alfabeto objetivo*** $\mathcal{T}=\{0,1\}$. Construyendo las ***extensiones del código***

$$C_{\text{LV}}=\{a\longmapsto 0,\ b\longmapsto 01,\ c\longmapsto 011\};$$
$$C_{\text{LF}}=\{a\longmapsto 00,\ b\longmapsto 01,\ c\longmapsto 11\};$$

el mensaje $abaaacab$ se ***codificada***

- En el código C_{LV} como 0 01 0 0 0 011 0 01 con una ***longitud de 12 bits***;
- En el código C_{LF} como 00 01 00 00 00 11 00 01 con una ***longitud de 16 bits***.

Muchas de las propiedades de los códigos (decodificación unívoca, prefijo, eficiencia, etc) están relacionadas con su longitud $l(C)$ y la longitud de las palabras códigos $l_w(C(s_i))$.

Esta relaciones están expresadas por la ***desigualdad de Kraft.***

Definición – Dada una fuente de q símbolos a codificar con un alfabeto de n símbolos utilizando un conjunto de q palabras de longitud $l_{w1}, \dots, l_{wq}$, respectivamente, entonces la desigualdad de Kraft está dada por

$$\sum_{s_i \in \mathcal{S}} n^{-l_{wi}} \leq 1$$

La desigualdad de Kraft limita las longitudes de palabras de código en un código: si se tiene una exponencial de la longitud de cada palabra de código válido, el conjunto resultante de valores es una distribución de valores positivoscon un valor medio inferior o igual a uno.
Esta desigualdad se puede pensar en términos de un presupuesto limitado para ser utilizado en palabras de código, donde palabras de código cortas son más caras y palabras código largas son más baratas.

Si la desigualdad de Kraft se cumple de manera ***estrictamente desigual***, el código tiene alguna ***redundancia***.

- **Distancia** – es una función matemática que mide la similitud entre dos palabras código t_i y t_j de un código que cumple con las siguientes propiedades
  - $d(t_i,t_j) = d(t_j,t_i)$;
  - $d(t_i,t_i) = 0$;
  - $d(t_i,t_j) + d(t_i,t_m) \geq d(t_i,t_m)$.

  Hay muchas definiciones de distancia para comparar secuencias (distancia de edición, distancia de Damerau-Levenshtein, distancia Manhattan, entre otras).

  La definición de distancia más utilizada en sistemas digitales y códigos es la ***distancia de Hamming***, que se la define como el número de posiciones en las que los símbolos correspondientes son diferentes.

  En otras palabras, mide el número mínimo de sustituciones necesarias para cambiar una cadena en la otra.

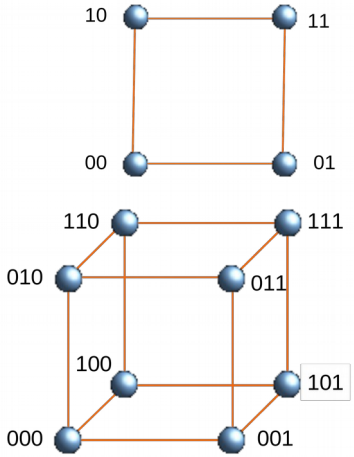
  La distancia Hamming entre **1011101** y **1001001** es 2.
  La distancia Hamming entre **2143896** y **2233796** es 3.
  La distancia Hamming entre "**tener**" y "**reses**" es 3.

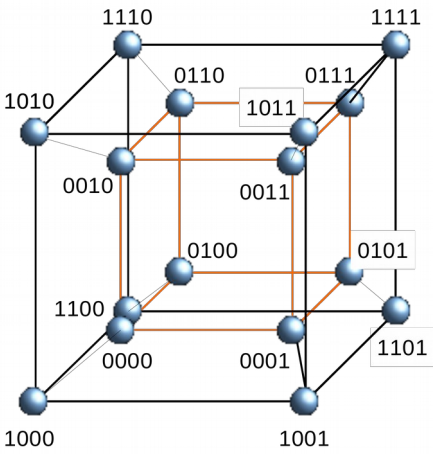
Uno de los usos más importantes de la distancia de Hamming es en teoría de la codificación, más específicamente para los códigos de bloque, en los que las cadenas de igual longitud son vectores sobre un campo finito.

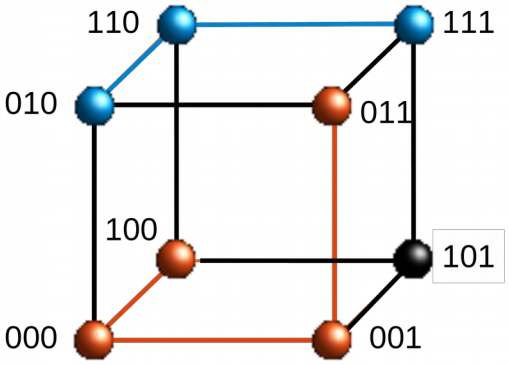
Para cadenas binarias s_1 y s_2, la distancia de Hamming es igual al número de unos (conteo de población) en una operación lógica

$$s_1 \text{ XOR } s_2.$$

El espacio métrico de las cadenas binarias de longitud r, con la distancia de Hamming, se conoce como el ***cubo de Hamming***; el cual es equivalente al conjunto de distancias entre vértices en un gráfico de hipercubo.







También se puede ver una cadena binaria de longitud n como un vector en $\mathbb{R}^n$ tratando cada símbolo en la cadena como una coordenada real. Con esta representación, las cadenas forman los vértices de un hipercubo r-dimensional, y la ***distancia de Hamming*** entre las cadenas es equivalente a la ***distancia de Manhattan*** entre los vértices.

- **Código continuo** – es un código en el cual la distancia de Hamming entre las ***palabras código es uno***.

- **Código cíclico** – es un código en el cual la ***última*** palabra código es ***continúa con la primera***.

- **Código autocomplementario** – es un códigoen el cual sus palabras códigos se complementan entre sí, es decir palabras código del conjunto objetivo se obtienen a partir de aplicar la operación negación a otras palabras código del conjunto.

- **Código completo** – es un código que utiliza todas las palabras código disponibles, es decir no hay redundancia lo que equivale a

$$\sum_{s_i \in \mathcal{S}} n^{-l_{wi}} = 1.$$

- Si $C: \mathcal{S} \longrightarrow \mathcal{T}$ es inyectiva, $s_i \neq s_j \Rightarrow t_i \neq t_j$, entonces C es invertible

  Un codigo es invertible si a cada $s_i \in \mathcal{S}$ lo mapea a un símbolo diferente $t \in \mathcal{T}$

- Si $C: \mathcal{S} \longrightarrow \mathcal{T}$ es inyectiva entonces C es ***decodificable de manera unívoca***

  Esto significa que a cada símbolo $s_i \in \mathcal{S}$ se le asigna una cadena de símbolos $t = C(s) \in \mathcal{T}$ diferente, esto implica que la extension del código es invertible.

Códigos – *Definiciones*

- **Código de prefijo** – es un código que tiene la ***propiedad de prefijo***, es decir no hay una palabra código que sea un prefijo (segmento inicial) de cualquier otra palabra del código.
- Es trivialmente cierto para el código de longitud fija, por lo que solo es un punto de consideración en el código de longitud variable.

- **Códigos instantáneos** – es un código que se puede decodificar cada palabra de código, en cualquier cadena, tan pronto como se recibe. Este tipo de códigos se pueden expresar mediante ***árboles de código***.

- Si $C: \mathcal{S} \longrightarrow \mathcal{T}$ es ***instantáneo*** si C no es un código prefijo
  Esto significa que el código se puede decodificar a medida que llegan las palabras código t, es los símbolos son decodificables de manera independiente.

- **Códigos óptimos** – es un código que es unívocamente decodificable y cuya longitud media es mínima (respecto a todos los demás códigos univocos sobre los mismos alfabetos fuente y código).

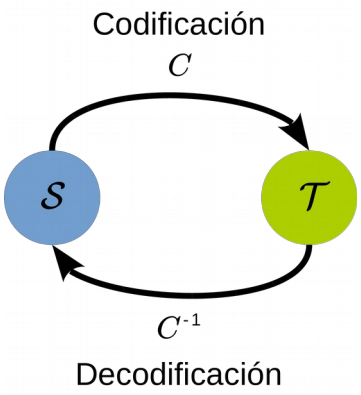
- **Códigos ponderados** – es un código en el cual hay una conexión aritmética entre el código, obedecen al principio de peso posicional: Cada posición del número representa un peso específico.

Decodificación – *Definición*

La ***decodificación*** es el proceso inverso por el cual los códigos se convierten a su forma original para ser utilizado y/o comprendido por el receptor.
Formalmente

Dado un mensaje $x \in \mathbb{F}_n^2$, entonces el ***decodificador ideal*** genera la palabra código $y \in C$ que es el resultado del proceso de toma de decisión

$$\mathbb{P}(y \text{ enviado} \mid x \text{ recibido})$$



Por ejemplo, una persona puede elegir la palabra de código y que es más probable que se reciba como mensaje x después de la transmisión.

Cada palabra de código no tiene una probabilidad esperada: puede haber más de una palabra de código con la misma probabilidad de mutar en el mensaje recibido. En tal caso, el emisor y el receptor deben acordar de antemano una convención de decodificación.
Las convenciones populares incluyen:

- Solicite que se reenvíe la palabra de código: solicitud de repetición automática.
- Elija cualquier palabra de código aleatorio del conjunto de palabras de código más probables que esté más cerca de eso.
- Si sigue otro código, marque los bits ambiguos de la palabra de código como borrados y espere que el código externo los desambigue.

Decodificación – *Máxima verosimilitud*

La **decodificación por máxima verosimilitud** maximiza la probabilidad del mensaje enviado $y \in C$ dado un mensaje recibido $x \in C$.

Dada la palabra código recibida $x \in C \in \mathbb{F}_n^2$, entonces el ***decodificador por máxima verosimilitud*** elije una palabra código $y \in C$ de modo que

$$max\ \mathbb{P}(x \text{ recibido} \mid y \text{ enviado})$$

es decir, la palabra de código y maximiza la probabilidad de que se haya recibido x dado que se envió y.

Si todas las palabras de código tienen la ***misma probabilidad de enviarse***, entonces este esquema es equivalente a la decodificación ***ideal del observador***.
Según el teorema de Bayes,

$$\mathbb{P}(x \text{ recibido} \mid y \text{ enviado}) = \frac{\mathbb{P}(x \text{ recibido}, y \text{ enviado})}{\mathbb{P}(y \text{ enviado})} = \mathbb{P}(y \text{ enviado} \mid x \text{ recibido}) \frac{\mathbb{P}(x \text{ recibido})}{\mathbb{P}(y \text{ enviado})}$$

Fijando $\mathbb{P}(x \text{ recibido})$ y como $\mathbb{P}(y \text{ enviado})$ es constante (porque las palabras son igualmente probables); entonces $\mathbb{P}(x \text{ recibido} \mid y \text{ enviado})$ se maximiza en función de y cuando $\mathbb{P}(y \text{ enviado} \mid x \text{ recibido})$ es maximo.

El problema de decodificación de máxima verosimilitud se puede modelar como un ***problema de optimización entera***.

Decodificación – *Mínima distancia*

Dada una palabra de código recibida $x \in \mathbb{F}_n^2$ la ***decodificación d distancia mínima*** selecciona una palabra de código $y \in C$ que minimice la distancia de Hamming $d(x,y)$ para todo $y \neq x$, es decir elijir la palabra código $y \in C$ que esté lo más cerca posible de $x \in C$.

Si la probabilidad de error en un canal sin memoria $p < 0{,}5$, entonces la decodificación de mínima distancia es ***equivalente*** a la ***decodificación de máxima verosimilitud***, ya que si $d(x,y) = d$, entonces la estimación de máxima verosimilitud esta dada por

$$\mathbb{P}(y \text{ recibido} \mid x \text{ enviado}) = (1-p)^{n-d} p^d = (1-p)^n \left(\frac{p}{1-p} \right)^d$$

que se maximiza minimizando d.

La ***decodificación de mínima distancia*** también se la conoce como ***decodificación de vecino más cercano*** (*nearest neighbour*).

La decodificación de mínima distancia se puede utilizar cuando se cumplen las siguientes condiciones:

- La probabilidad de que ocurra un error p es ***independiente de la posición*** del símbolo; y
- Los errores son eventos ***independientes***: un error en una posición del mensaje ***no afecta a otras posiciones***.

Decodificación – *Sindrome*

La ***decodificación del síndrome*** es la decodificación de distancia mínima utilizando una tabla de búsqueda reducida. Esto se puede hacer por la propiedad de linealidad de los códigos lineales.

Supongamos que $C \subset \mathbb{F}_n^2$ es un código lineal de longitud n, distancia mínima de Hamming d y matriz de chequeo de paridad H, entonces es capaz de corregir hasta

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor$$

Dada una palabra de código $x \in \mathbb{F}_n^2$ enviada y se le agrega el error $e \in \mathbb{F}_n^2$ tal que $z = x + e$ es recibida.

La decodificación de mínima distancia busca el vector $y \in C$ en una tabla de tamaño $|C|$ la coincidencia más cercana, un elemento (***no necesariamente único***) $c \in C$ con $d(c,z) \leqslant d(y,z)$ para todo $y \in C$.

La decodificación del síndrome aprovecha la propiedad de la matriz de paridad que

$$H\,x = 0 \text{ for all } x \in C.$$

El síndrome de $z = x + e$ se define como

$$H\,z = H(x+e) = Hz = Hx + He = 0 + He = He.$$

Conociendo e es trivial decodificar x como $x = z - e$.

Para realizar la decodificación, se busca en una tabla precalculada de tamaño 2^{n-k} que asigne He a e.

Bajo el supuesto de que ***sólo se cometen t errores***, el decodificador busca el valor He en una tabla de tamaño reducido

$$\sum_{i=0}^{t} \binom{n}{i} < |C|$$

Decodificación – *Matriz estandar*

La ***decodificación por matriz estandar*** utiliza una matriz $q^{n-k} \times q^k$ que enumera los elementos de un $[n,k]$-código lineal $C \in \mathbb{F}_q^n$.

La matriz estandar se utiliza para encontrar la palabra código que corresponde a la palabra recibida donde

- La primera fila enumera todas las palabras de código;
- Cada fila es un coset con el líder del coset en la primera columna;
- La entrada i-ésima fila y j-ésima columna es la suma del i-ésimo líder de coset y la j-ésima palabra código.

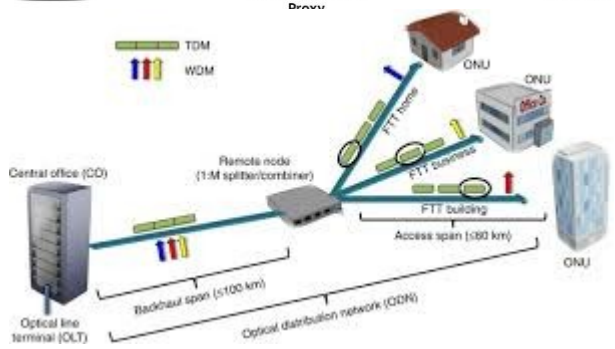
Una matriz estándar se construye de la siguiente manera

1. Liste las palabras de código de C comenzando con 0 en la primera fila;
2. Elija cualquier palabra de peso mínimo que aún no esté en la matriz. Escribala como la primera entrada de la siguiente fila. Este vector se denota como el "*líder de coset*".
3. Complete la fila agregando el líder de coset a la palabra de código en la parte superior de cada columna. La suma del líder del coset i-ésimo y la palabra de código j-ésima es la entrada en la fila i, columna j.
4. Repita los pasos 2 y 3 hasta que se enumeren todos los cosets y cada palabra aparezca una vez.

Para decodificar una palabra se le resta el líder de coset, el resultado será una de las palabras de código.
La decodificación por matriz estándar es una forma de decodificación de vecino más cercano.
En la práctica, la decodificación a través de una matriz estándar requiere grandes cantidades de almacenamiento y garantiza que los vectores sean correctamente decodificados.

Hay cuatro usos básicos de la codificación

- **Compresión de datos** (o codificación de fuente) – eliminan la redundancia de los datos de una fuente para transmitirla de manera más eficiente. La compresión de datos y la corrección de errores pueden estudiarse en combinación;
- **Control de errores** (o codificación de canal) – agrega redundancia (algunos datos adicionales) a un mensaje, de modo que los receptores pueden usarla para verificar la consistencia del mensaje entregado y para recuperar datos que se ha determinado que están dañados;
- **Codificación criptográfica** – se trata de construir y analizar protocolos que eviten que terceros accedan a información privada. Varios aspectos de la seguridad de la información, como la confidencialidad de los datos, la integridad de los datos, la autenticación y el no repudio son temas abordados por esta codificación;



- **Codificación de línea** (o transmisión de banda base) se utiliza para representar la señal digital a transportar por una señal analógica que cambia alguna de sus propiedades (amplitud, frecuencia y fase) en el tiempo de manera óptima para las propiedades específicas del canal físico (y del equipo receptor).