FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Maestría en Sistemas Embebidos

Sistemas Digitales para las Comunicaciones

Sistemas de comunicación

El punto de vista de la teoría de la información

Parte 4

Cronograma

Parte 0

Parte 1

Parte 2

Parte 3

Parte 4

Parte 5

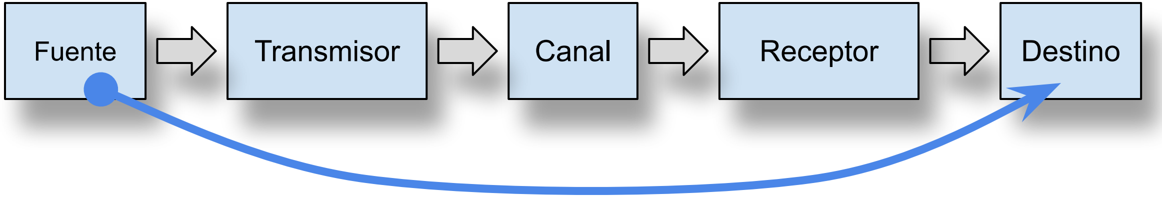
Parte 6

Parte 4: Sistema de comunicación - Teoría de información.

- Introducción a los sistemas de comunicación digitales.
  - Fuente, mensaje, transmisor, receptor y canal.
  - Codificación.
- Codificación de fuente.
  - Fuentes: Información y propiedades.
  - Codificación de Huffman y Lempel-Ziv.
- Codificación de canal.
  - Modelos y capacidad de canal.
  - Detección y corrección de errores.
  - Códigos por bloque lineales y convolucionales.
  - Entrelazado y codificación de línea.

Sistema de comunicación:

- **Fuente de mensaje**
- **Transmisor**
- **Canal**
- **Receptor**
- **Destino del mensaje**

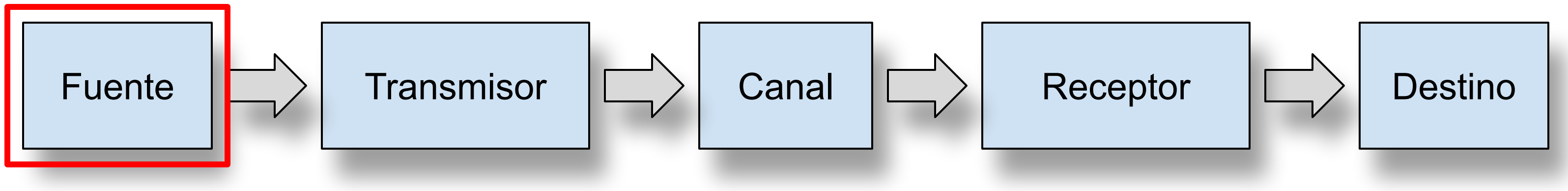


Mensaje

- Existe un **mensaje** que se genera en la **fuente**.
- El mensaje es aleatorio.
- Se quiere enviar al **destino** …
- … a través de un **canal**.
- Es **transmisor** se encarga de adaptar el mensaje para que atraviese el canal.
- El **receptor** se encarga de recuperar el mensaje y volverlo a su formato original.

Sistema de comunicación

Fuente de mensaje

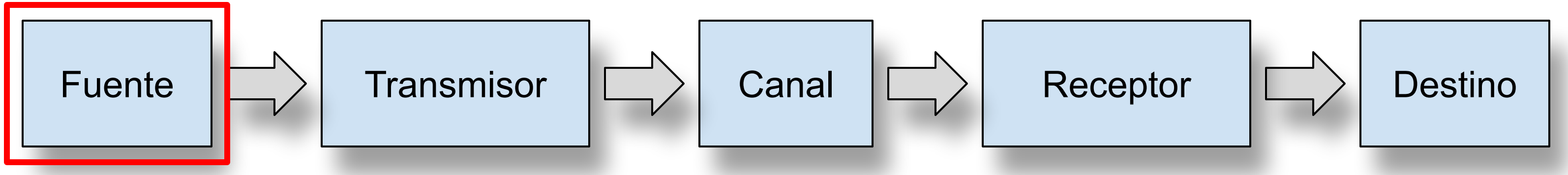


Emite un mensaje que es **aleatorio**.

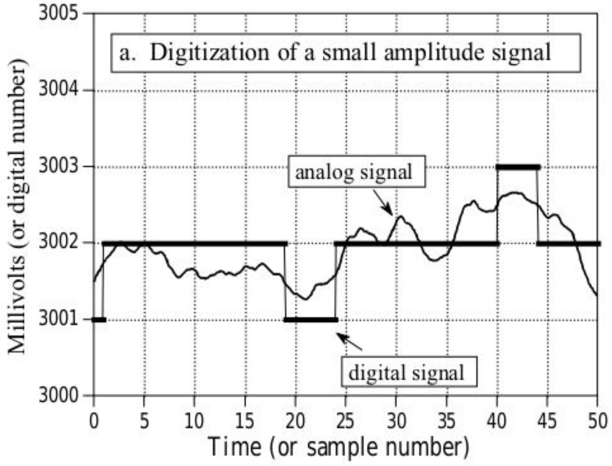
Si no fuese aleatorio no habría “nada para transmitir”.

Sistema de comunicación

Fuente de mensaje



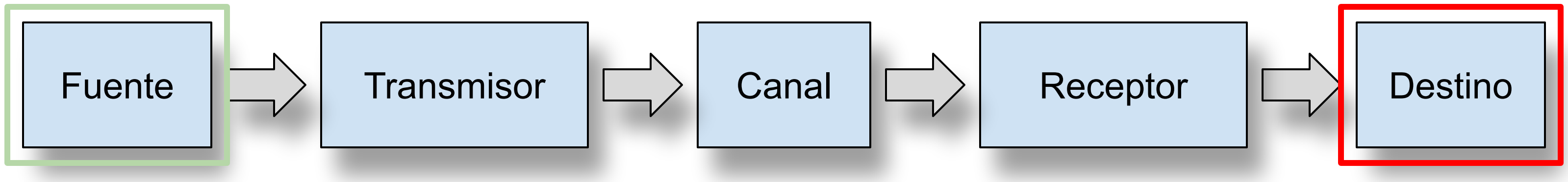
- Fuentes **discretas** (en tiempo y amplitud):
  - Tener un alfabeto
  - Tener una tasa de símbolos R



- Fuente **continua**:
  - **Cuantización** y **muestreo**
  - En general con **pérdida** de información: Teoría de Rate-Distortion

Sistema de comunicación

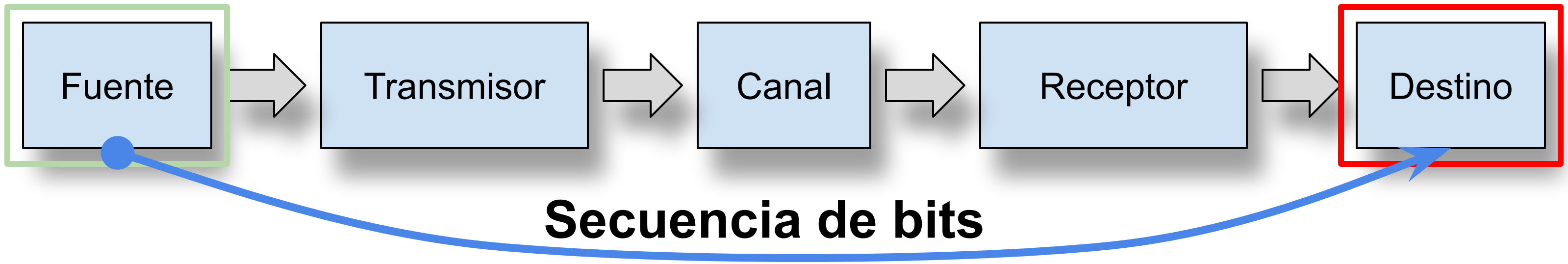
Destino



“Consume” el mensaje en el formato original.

Sistema de comunicación

Sistema de comunicación digital

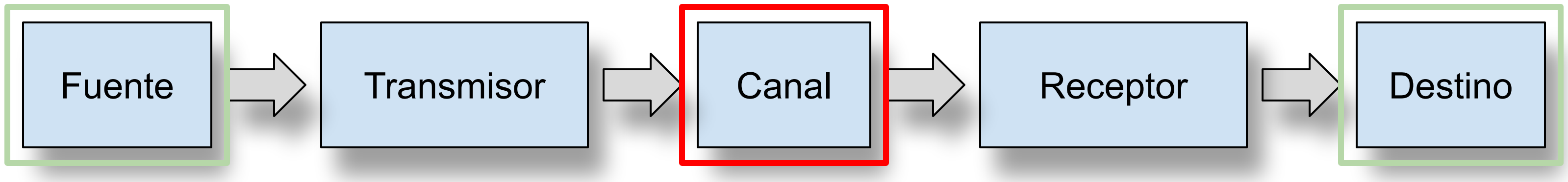


A partir de ahora consideramos que tenemos un mensaje discreto:

- Si tengo fuente analógica la cuantizo y muestreo
    - Me olvido de la distorsión y pérdida de información: Problema de otro área
- Alfabeto con $M = 2^b$ letras, equivalente a agrupar bits
    - Mensaje = Secuencia de bits
- Entonces tengo un sistema de **comunicación digital**

Sistema de comunicación

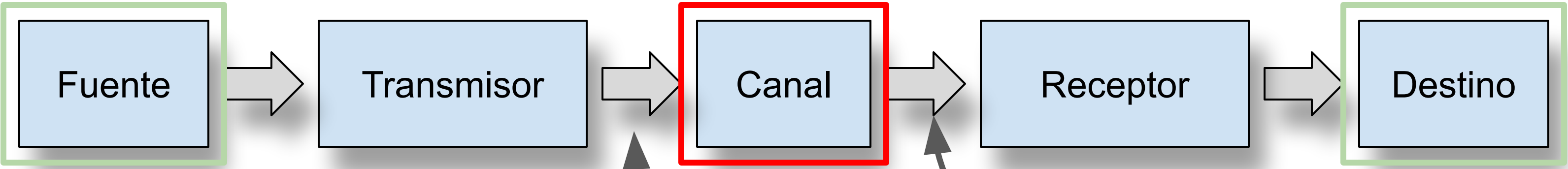
Canal



- El canal es un medio inherentemente **analógico**
- Introduce **distorsiones** en el mensaje que se quiere transmitir
  - Respuesta en frecuencia
  - Atenuación
  - Ruido
  - Otras fuentes de distorsión lineales y no lineales

Sistema de comunicación

Canal

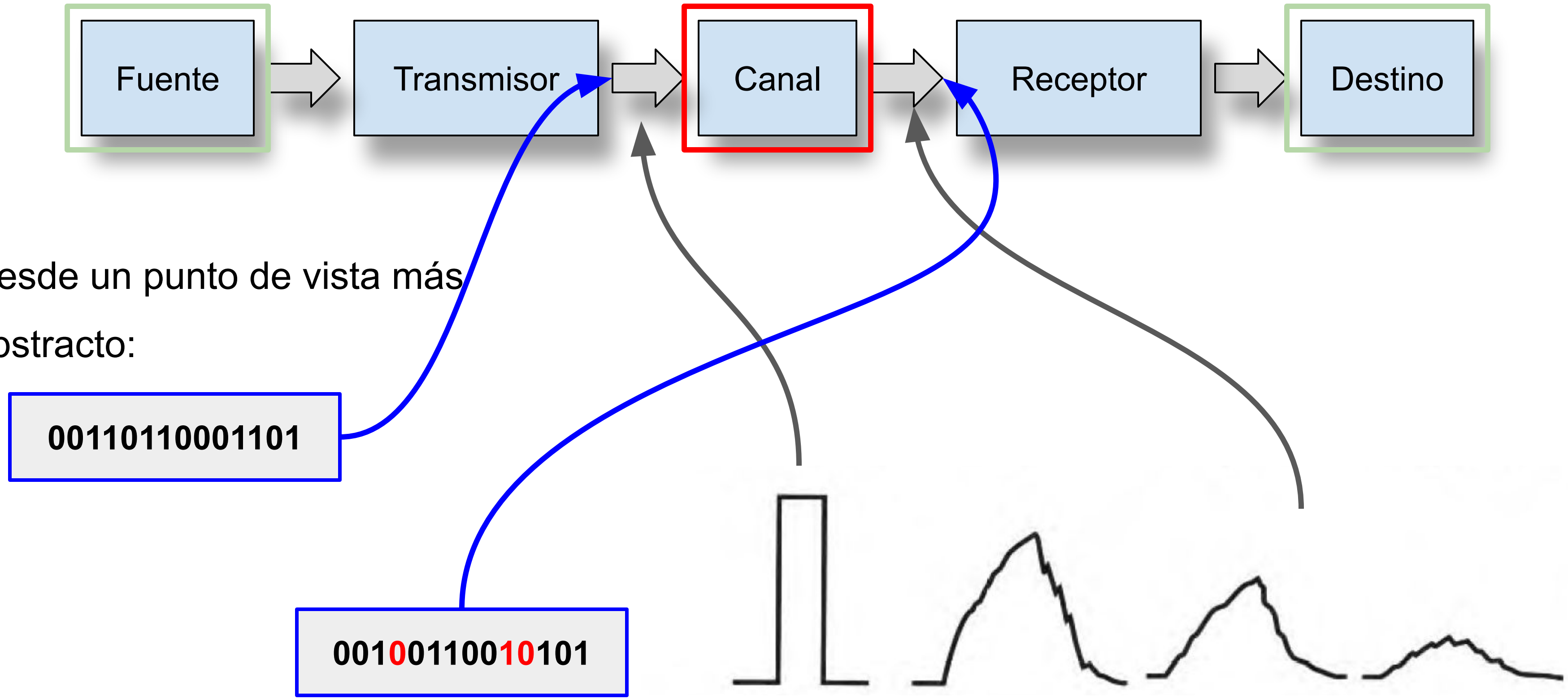


- El canal es un medio inherentemente **analógico**
- Introduce **distorsiones** en el mensaje que se quiere transmitir
  - Respuesta en frecuencia
  - Atenuación
  - Ruido
  - Otras fuentes de distorsión lineales y no lineales

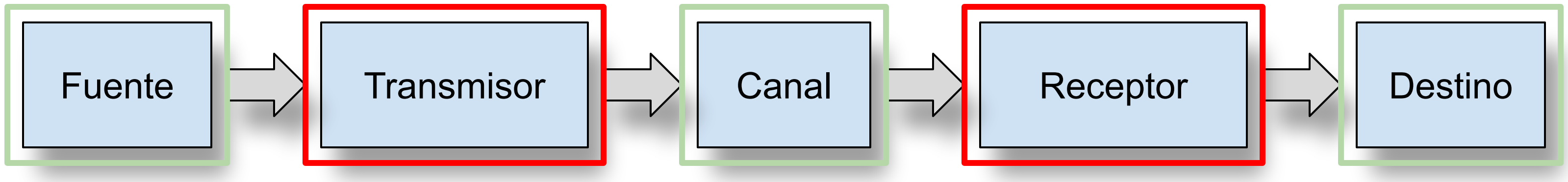
Sistema de comunicación

Canal

Desde un punto de vista más abstracto:



Capacidad del canal



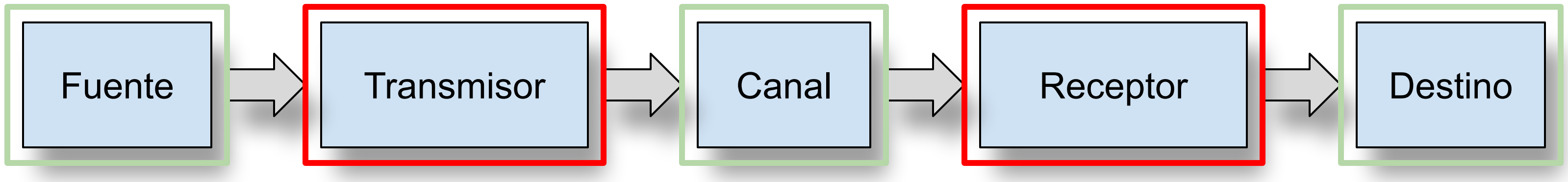
¿Qué hacen el transmisor y el receptor?

Hacen posible el envío del mensaje

¿Cómo?

Sistema de comunicación

Capacidad del canal

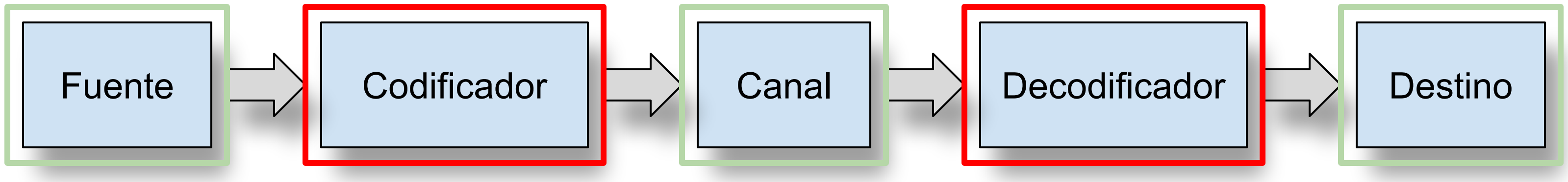


Objetivo del transmisor y receptor en conjunto:

- Extraer la información de la fuente
- Revertir los errores introducidos por el canal

Sistema de comunicación

Transmisor y receptor



Objetivo del transmisor y receptor en conjunto:

- Extraer la información de la fuente
- Revertir los errores introducidos por el canal

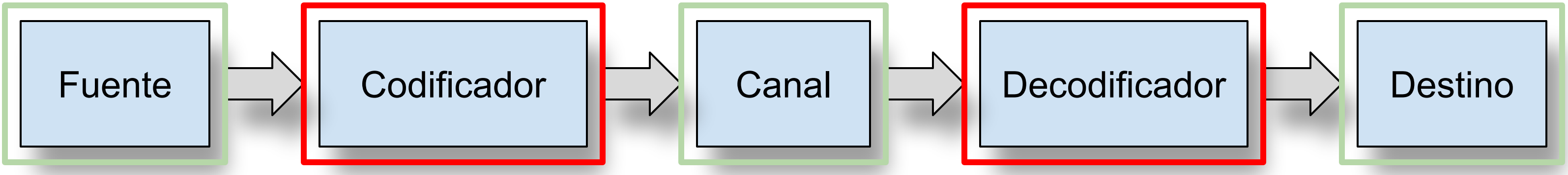
El proceso para alcanzar ambos los objetivos se llama **codificación**

Se puede ver intuitivamente que es un proceso de dos etapas.

- Codificación de fuente
- Codificación de canal

Sistema de comunicación

Transmisor y receptor



Objetivo del transmisor y receptor en conjunto:

- Extraer la información de la fuente
- Revertir los errores introducidos por el ca

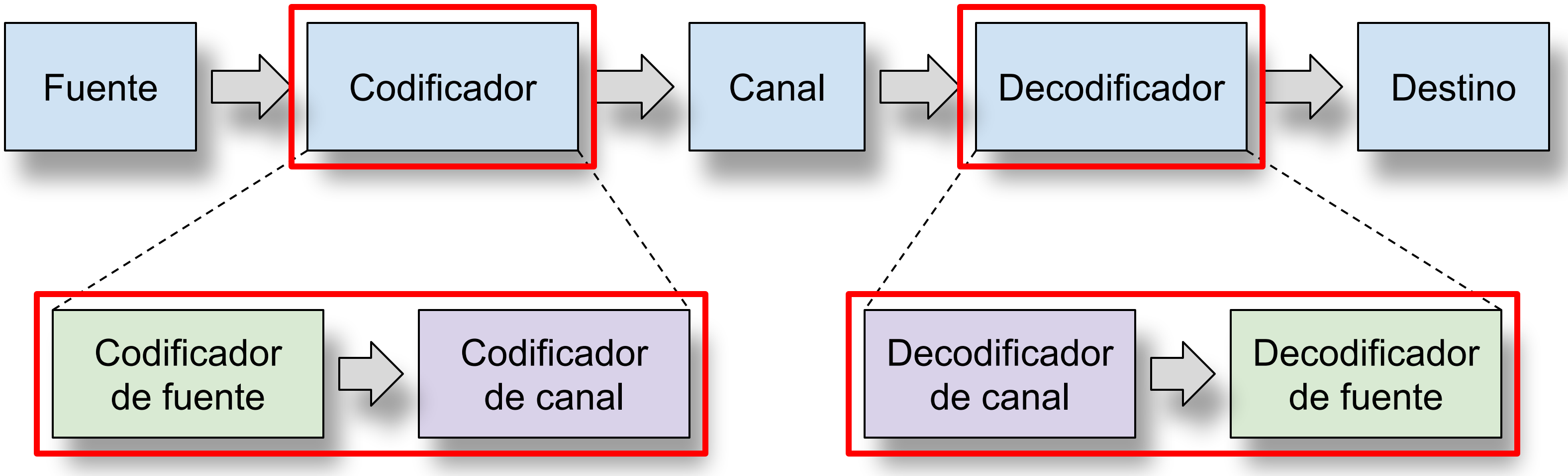
Se puede demostrar que las dos etapas son independientes. Es un resultado importante de la teoría de información: "*source-channel separation theorem*"

El proceso para alcanzar ambos los objetivos se llama **codificación**

Se puede ver intuitivamente que es un proceso de dos etapas.

- Codificación de fuente
- Codificación de canal

Transmisor y receptor

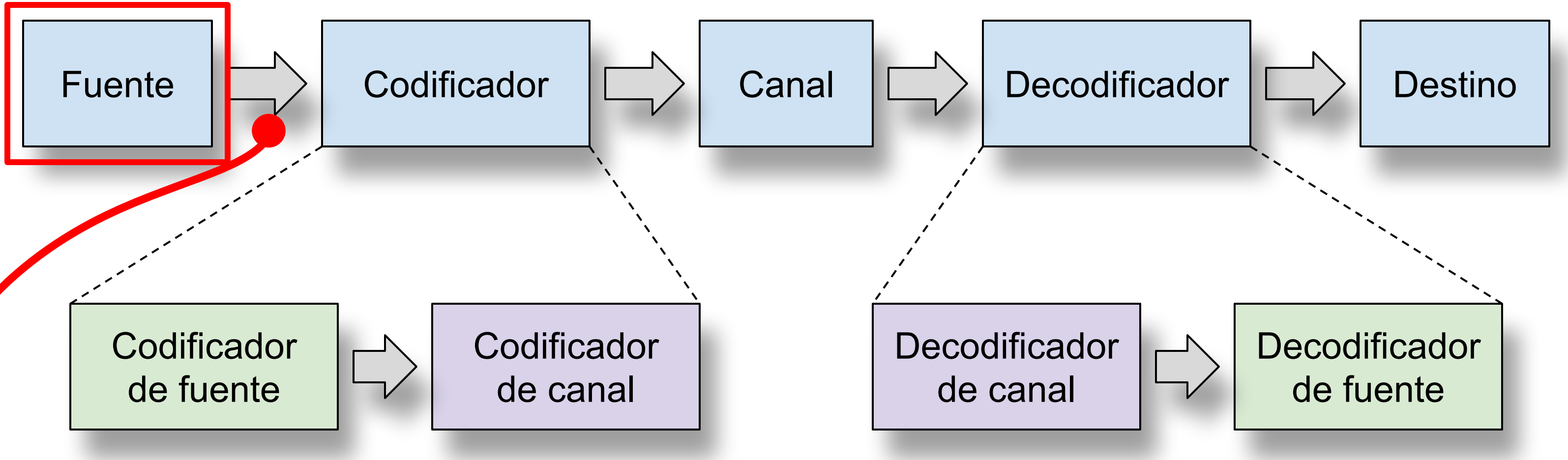


Se divide el problema en dos subproblemas pueden ser atacados de manera independiente:

[De]codificación de fuente y [De]codificación de canal

Sistema de comunicación

Resumen



Fuente y alfabeto de la fuente

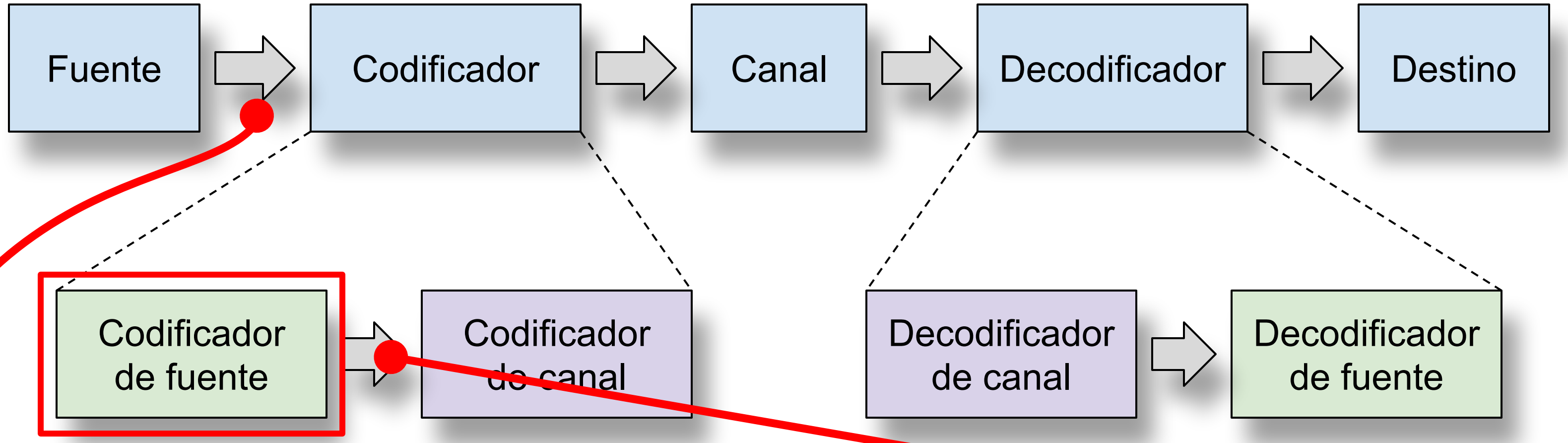
Puede tener muchos datos y poca información

Puede tener pocos datos y mucha información

Es necesaria una medida de información

Sistema de comunicación

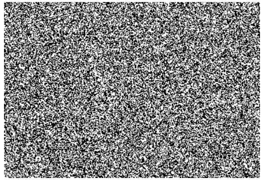
Resumen



Codificador de fuente

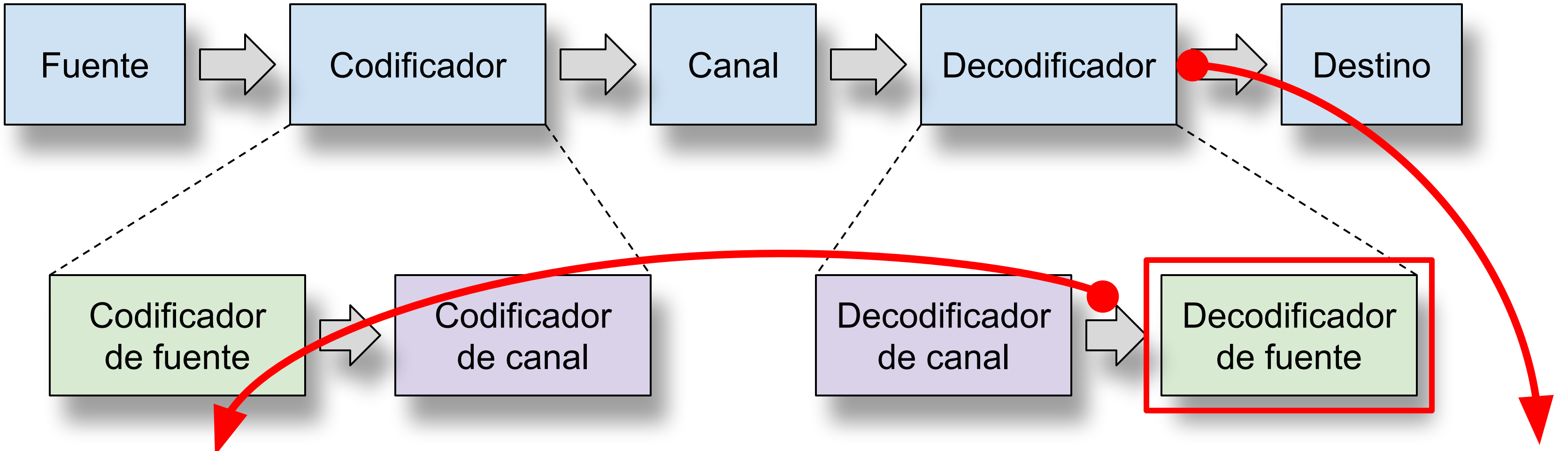
Extrae la información, comprime los datos

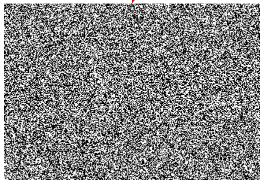
La salida es un alfabeto binario “muy aleatorio”, sin redundancias



Sistema de comunicación

Resumen





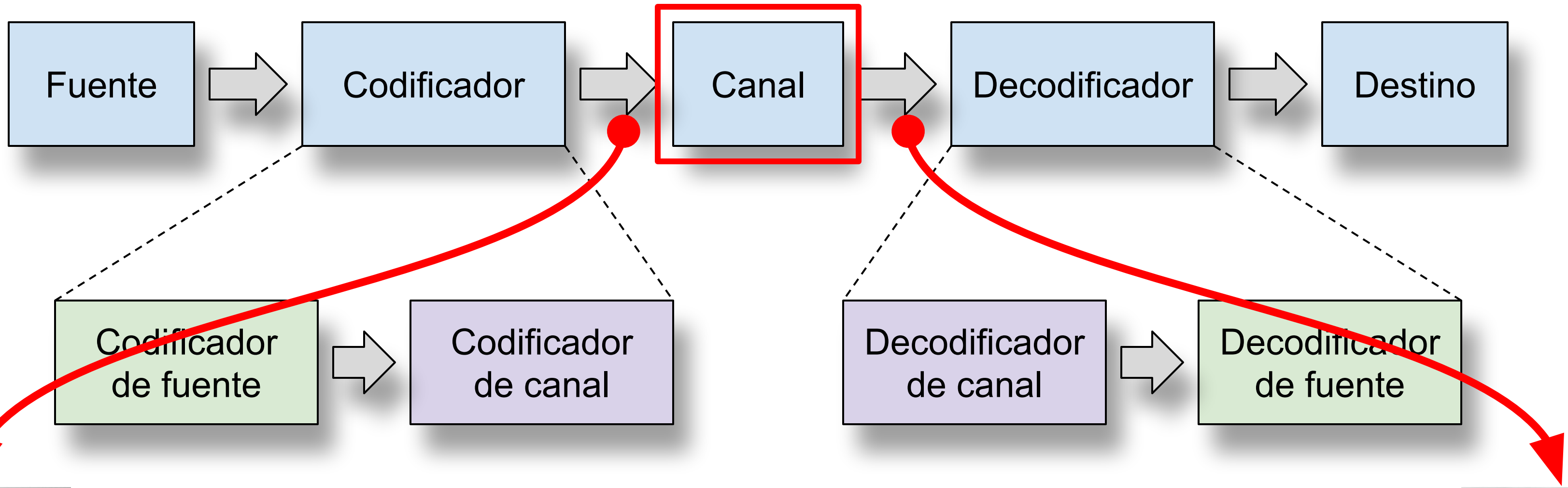
Decodificador de fuente

Descomprime los datos

Revierte el proceso de codificación

Presenta los datos de manera conveniente

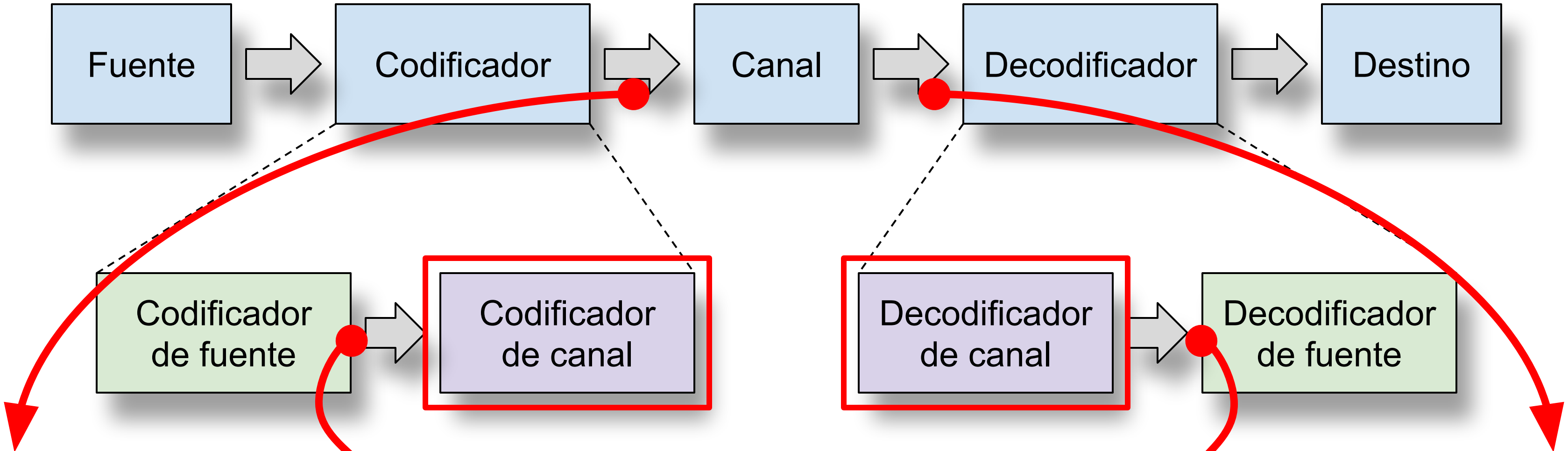
Resumen



Canal

Introduce distorsiones: Ruido, atenuación y respuesta en frecuencia, lo cual genera errores en los datos.

Resumen



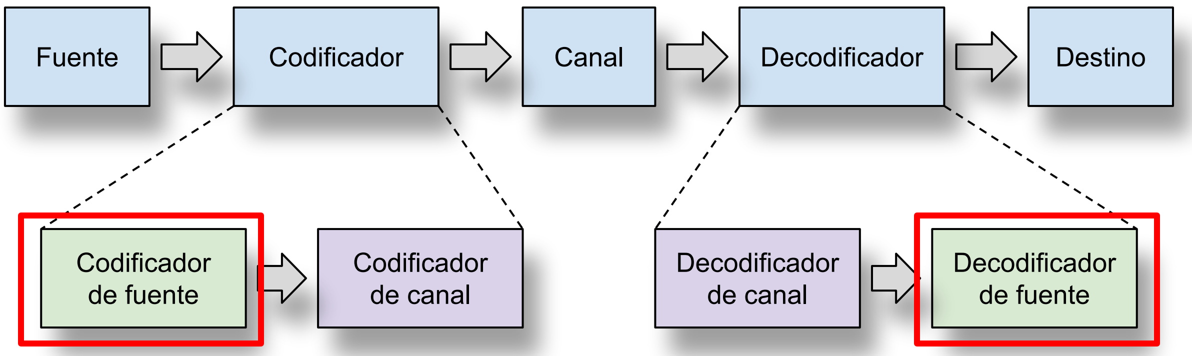
Codificación y decodificación de canal

Agrega redundancia

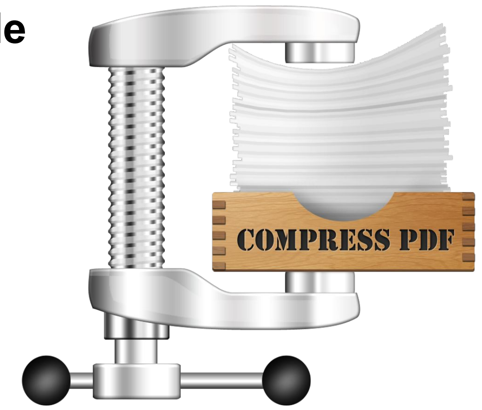
Revierte los errores introducidos por el canal

Codificación de fuente:

- **Introducción a la teoría de información**
  - **Medida de información**
  - **Modelos de fuentes**
- **Códigos**
  - **De largo variable**
  - **Cod. de Huffman**
  - **Compresión Lempel-Ziv**



El objetivo principal es representar los datos con la menor cantidad posible de bits. Esto se llama **compresión de datos**.



Codificación de fuente

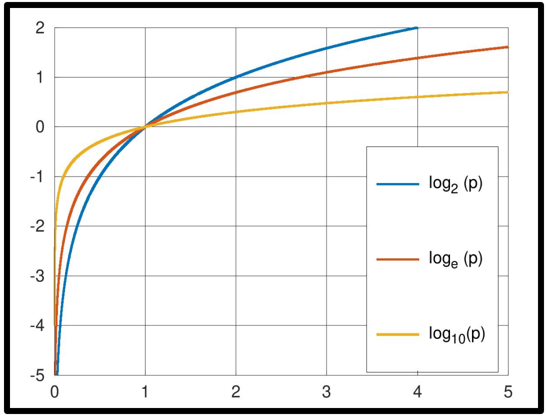
Medida de información

Consideremos las siguientes titulos de una noticia:

“*El sol salió esta mañana por el este en nuestro país*”

“*El COVID-19 se extinguió por completo en el mundo*”

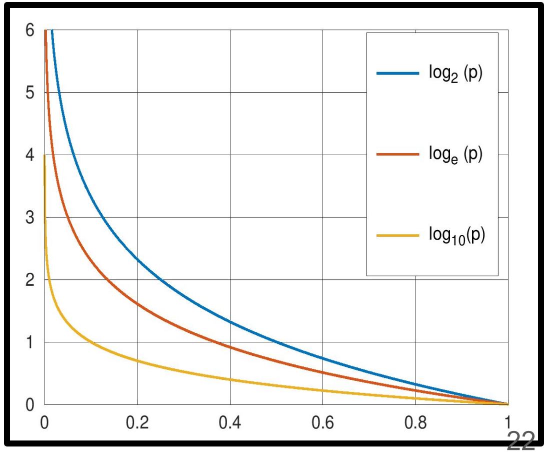
“*Se comprobó que el COVID-19 nunca existió*”



Es una **secuencia de letras de un alfabeto**

- ¿Cuál de estas atrae más la atención del lector?
- Evaluemos las probabilidades de cada uno de esos eventos
- ¿Existe una relación?

  Si **Prob=1 ⇒ Info=0**, mientras que si **Prob=0 ⇒ Info=∞**
- Matemáticamente una posible medida es: **Info = $\log_b$(1/Prob)**



Modelos de fuentes discretas

Es una **secuencia de letras de un alfabeto**

- Alfabeto: Un conjunto de letras
- Fuente: Secuencia $\{X_1, X_2, X_3, \ldots, X_L\}$ de letras
- Prob. de cada letra

Ejemplo binario:

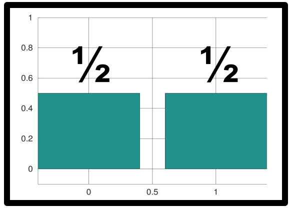
- $L = \{0,1\}$
- $P(X_i=0) = p$, $P(X_i=1) = 1-p$

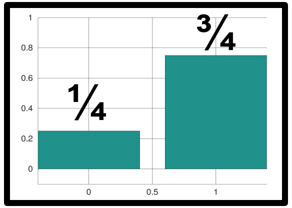
Ejemplo del idioma español:

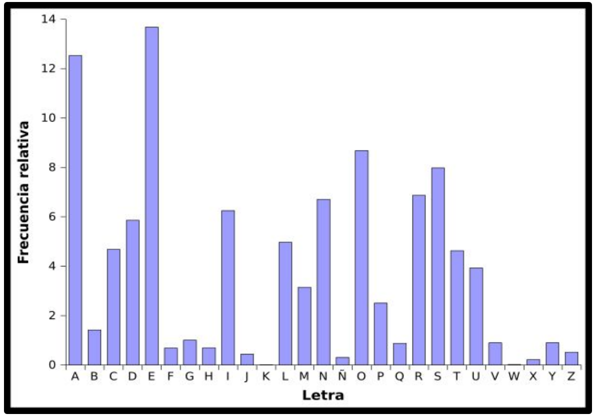
- $L = \{a, b, \ldots, z\}$
- $P(X_i=a) = 0.1253\ldots$

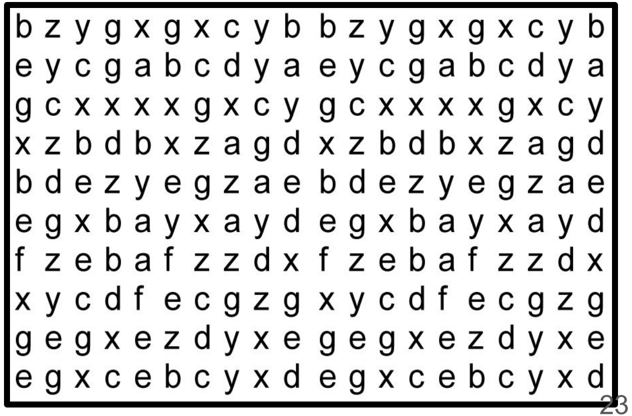
  $P(X_i=b) = 0.0142\ldots$

  ...





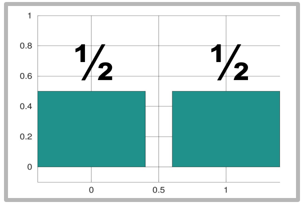




Codificación de fuente

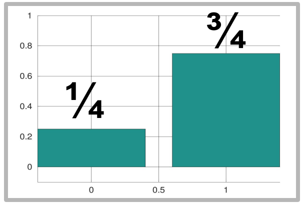
Entropía de una fuente

Información de …



$M = \{0101...01010\}$

$M = \{X_1, X_2, ..., X_L\}$



$M = \{0101...01010\}$

$M = \{X_1, X_2, ..., X_L\}$

… cada **símbolo**

$I(X=0) = -\log_2(p) = 1$

$I(X=1) = -\log_2(1-p) = 1$

$I(X=0) = -\log_2(p) = 2$

$I(X=1) = -\log_2(1-p) = 0.415...$

… una **letra**: Prom. información (E[I])

$H(X) = \text{Promedio}(I) =$

$= \frac{1}{2} (-\log_2(\frac{1}{2})) + \frac{1}{2} (-\log_2(\frac{1}{2})) =$

$= \frac{1}{2}\, 1 + \frac{1}{2}\, 1 = 1$

$H(X) = \text{Promedio}(I) =$

$= \frac{1}{4} (-\log_2(\frac{1}{4})) + \frac{3}{4} (-\log_2(\frac{3}{4})) =$

$= \frac{1}{4}\, 2 + \frac{3}{4}\, 0.415\ldots = 0.811...$

… un **mensaje** (secuencia de letras) es:

$\mathbf{H(M) =} H(\{X_1, X_2, ..., X_L\}) =$

$= H(X_1) + H(X_2) + \ldots + H(X_L) =$

$= L * H(X) \mathbf{= L}$

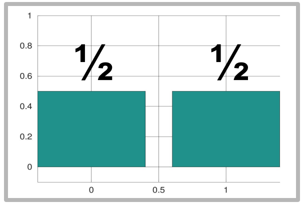
$\mathbf{H(M) =} H(\{X_1, X_2, ..., X_L\}) =$

$= H(X_1) + H(X_2) + \ldots + H(X_L) =$

$= L * H(X) \mathbf{= 0.81\ L}$

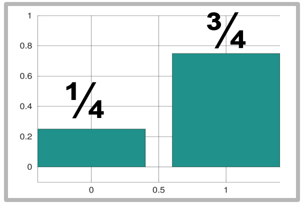
Entropía de una fuente

Información de …



M = {0101...01010}

M = $\{X_1, X_2, \ldots, X_L\}$



M = {0101...01010}

M = $\{X_1, X_2, \ldots, X_L\}$

… cada **símbolo**

… una **letra**: Porm. información

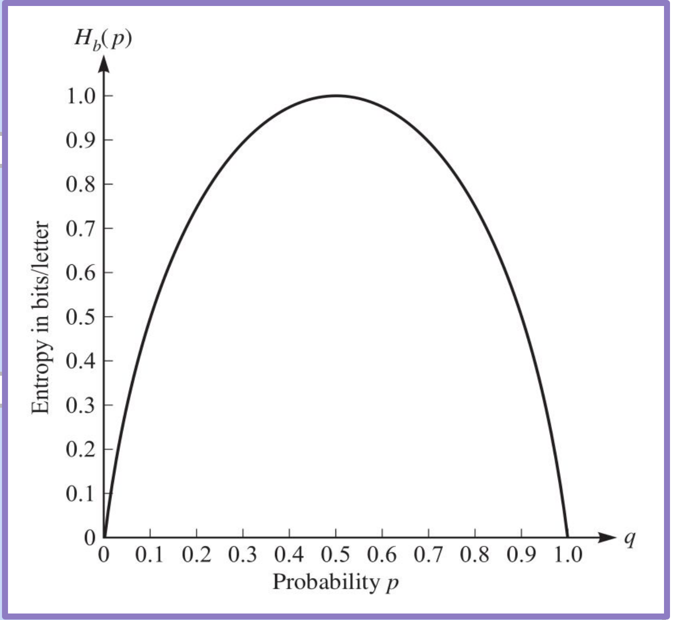
… un **mensaje** (secuencia de letras) es:

La entropía de una letra es el promedio de la información. Se mide en bits (cuando se usa $\log_2$)

Un mensaje es un conjunto de *L* letras y su entropía es.

Para una fuente **sin memoria**, **estacionaria** y **discreta** vale que:

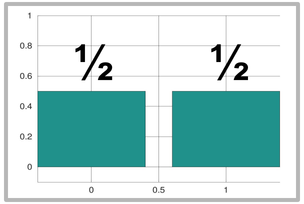
***H(X)* : (ver gráfico) [bits]**
***H(M)* = *L H(X)* [bits]**



Codificación de fuente

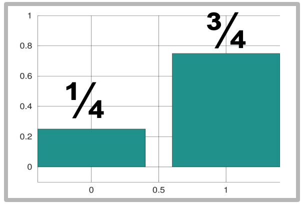
Entropía de una fuente

Información de …



$M = \{0101...01010\}$

$M = \{X_1, X_2, ..., X_L\}$



$M = \{0101...01010\}$

$M = \{X_1, X_2, ..., X_L\}$

… cada **símbolo**

… una **letra**: Porm. información

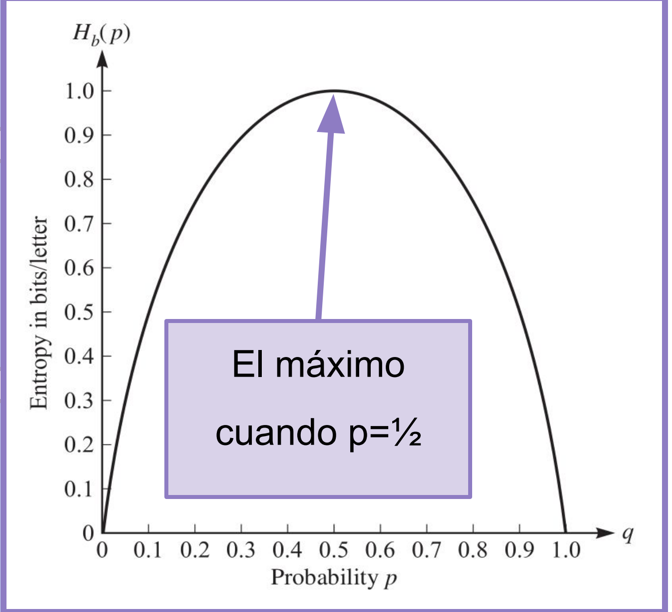
… un **mensaje** (secuencia de letras) es:

La entropía de una letra es el promedio de la información. Se mide en bits (cuando se usa $\log_2$)

Un mensaje es un conjunto de *L* letras y su entropía es.

Para una fuente **sin memoria**, **estacionaria** y **discreta** vale que:

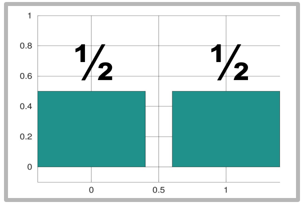
H(X) **: (ver gráfico) [bits]**
H(M) **=** ***L H(X)*** **[bits]**



Codificación de fuente

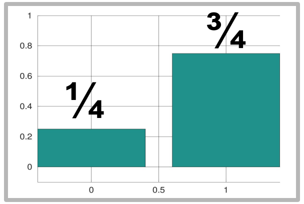
Entropía de una fuente

Información de …



M = {0101...01010}

$M = \{X_1, X_2, ..., X_L\}$



M = {0101...01010}

$M = \{X_1, X_2, ..., X_L\}$

… cada **símbolo**

… una **letra**: Porm. información

… un **mensaje** (secuencia de letras) es:

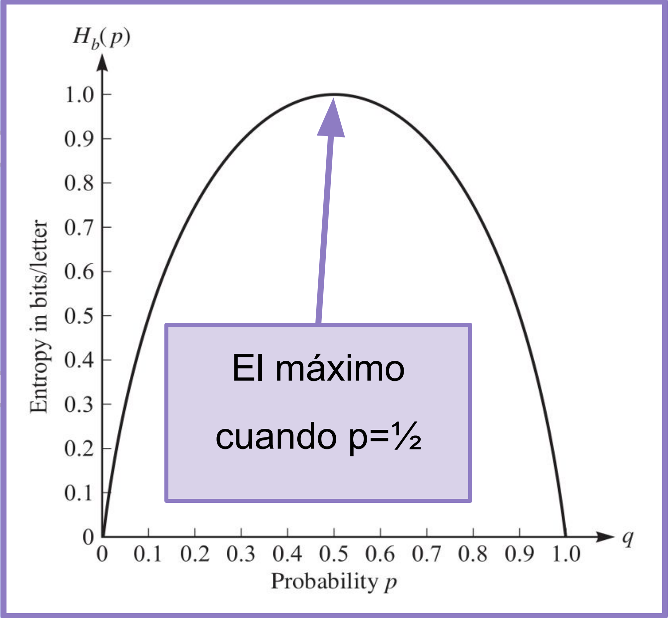
La entropía de una letra es el promedio de la información. Se mide en bits (cuando se usa $\log_2$)

Un mensaje es un conjunto de *L* letras y su entropía es.

Para una fuente **sin memoria**, **estacionaria** y **discreta** vale que:

¿?

H(X) **: (ver gráfico) [bits]**
H(M) **=** ***L H(X)*** **[bits]**



Memoria de una fuente:

Si la probabilidad de las letras es independiente del pasado, se dice que el canal **no tiene memoria**

Ejemplo fuente binaria:

010010101001010101011101100110101011000
110101111010101001010100110010101101011

000000000111111111111111111110000000000
000000001111111111111110000000111110000

Ejemplo idioma español:

b z y g x g x c y b
e y c g a b c d y a
g c x x x x g x c y
x z b d b x z a g d
b d e z y e g z a e
e g x b a y x a y d
f z e b a f z z d x
x y c d f e c g z g
g e g x e z d y x e
e g x c e b c y x d

HOLA COMO ESTAS YO ESTOY BIEN POR SUERTE

HOLA MUNDO

HOLA OLA VASO CAMA UVA PURE CASA PLACA LAPIZ

Modelos de fuentes discretas

Memoria de una fuen

Si la probabilidad de l

independiente del pasado, se dice que el

canal **no tiene memoria**

¿Cual tiene menos memoria?

Ejemplo fuente binaria:

010010101001010101011101100110101011000
110101111010101001010100110010101101011

000000000111111111111111111110000000000
000000001111111111111110000000111110000

Ejemplo idioma español:

b z y g x g x c y b
e y c g a b c d y a
g c x x x x g x c y
x z b d b x z a g d
b d e z y e g z a e
e g x b a y x a y d
f z e b a f z z d x
x y c d f e c g z g
g e g x e z d y x e
e g x c e b c y x d

HOLA COMO ESTAS YO
ESTOY BIEN POR SUERTE

HOLA MUNDO

HOLA OLA VASO CAMA UVA
PURE CASA PLACA LAPIZ

Memoria de una fuen

Si la probabilidad de l

independiente del pasado, se dice que el

canal **no tiene memoria**

¿Cual tiene menos memoria?

Ejemplo fuente binaria:

```
01001 1010010101 1011101100110101 1000
110101111 1010100101 0011001010 01011
```

```
00000 0001111111 1111111111100000 0000
000000001 1111111111 1000000011 10000
```

Ejemplo idioma español:

```
b   y g x g x c y b
e y c g a b c d y a
g c   x x x   x c y
x z b d b x z a g d
b d e z y e g z a e
e g x   a y x a y d
f z e b a f z z d x
x y c d f e c g z g
  e g x e z d   x e
e g x c e b c y x d
```

HOL C MO ESTAS YO
STOY BIEN P R SUERTE

HOL M NDO

OLA O A VASO CAM U A
PURE ASA PLAC LAPIZ

Memoria de una fuen

Si la probabilidad de la

independiente del pasado, se dice que el

canal **no tiene memoria**

¿Cual tiene menos memoria?

Ejemplo fuente binaria:

```
01001 1010010101 1011101100110101 1000
110101111 1010100101 0011001010 01011
```

```
00000 0001111111 1111111111100000 0000
000000001 11111111111 1000000011 10000
```

En general la memoria de una fuente reduce su entropía (información):

$$H(M_1) = L\,H(X) > H(M_2)$$

M_1: Fuente sin memoria
M_2: Fuente con memoria

Estacionariedad:

Sus propiedades estadísticas no cambian en el tiempo.

```
0100101010011100011010111101010100101010011001010110101101010101110110001101011110
1010100101010011001010110101111001101010110001101011110101011000110101111010101001
0101001100101011010111001010100110010110001101011110101010010101001100101011010111
```

```
0110100010001110010101010100010001010010101000001000001001000010100000100000100100
0100010000101000000001000000000100000000100100000000000000100000010000000000000000
0000000000010000000000001000000000000000000000000001000000000000000000000000001000000
```

Modelos de fuentes discretas

Estacionariedad:

Sus propiedades estad

¿Qué fuente es estacionaria?

0100101010011100011010111101010100101010011001010110101101010101110110001101011110
1010100101010011001010110101111001101010110001101011110101011000110101111010101001
0101001100101011010111001010100110010110001101011110101010010101001100101011010111

0110100010001110010101010100010001010010101000001000001001000010100000100000100100
0100010000101000000000100000000010000000010010000000000000100000010000000000000000
0000000000010000000000001000000000000000000000000000000010000000000000000000000001000000

Estacionariedad:

Sus propiedades estad

¿Qué fuente es estacionaria?

```
0100101010011100011010111101010100101010011001010110101101010101110110001101011110
1010100101010011001010110101111001101010110001101011110101011000110101111010101001
010100110010101101011100101010011001011000110101111010101001010100110010101101011 1
```

```
011010001000111001010101010001000101001010100000100000100100001010000010000001001 00
0100010000101000000000100000000010000000010010000000000000010000001000000000000000000
000000000001000000000000010000000000000000000000000000001000000000000000000000000001000000
```

Está relacionado con el concepto de **invarianza en el tiempo**.

RECORDEMOS:

El objetivo de la *codificación de fuente* es minimizar el número de bits necesarios para representar los mensajes (**compresión**).

IMPORTANTE:

Cuando hablamos de fuentes binarias el “**bit**” se utiliza como componente mínimo de una letra (letras compuestas por un conjunto de bits) y también como unidad para la entropía.

Codificación de fuente

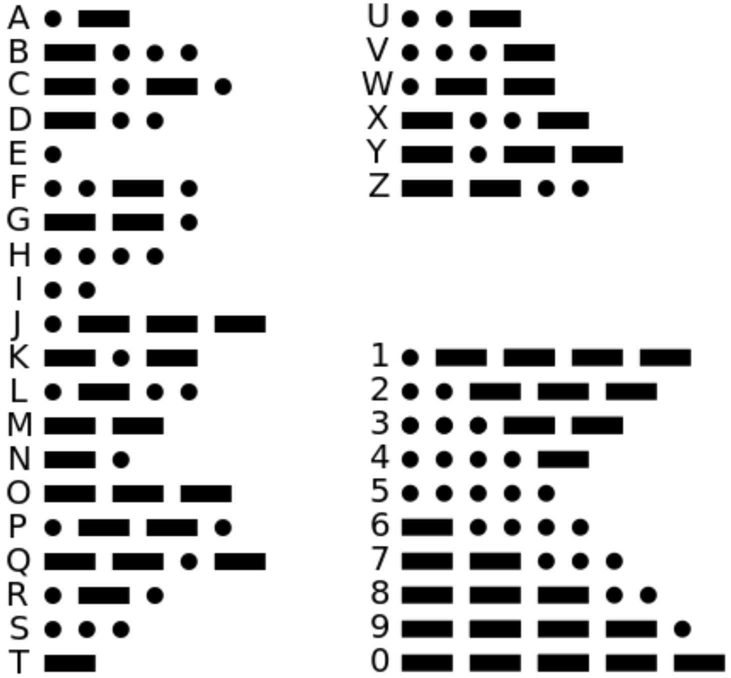
Códigos

Códigos de largo variable:

Son útiles cuando las letras de la fuente no son equiprobables. La estrategia para reducir el número de bits es:

- Evaluar la probabilidad de cada letra.
- Asignar los códigos más cortos a las letras más probables y los códigos más largos a las letras menos probables.

Un ejemplo es el código Morse.



Códigos de largo variable:

Otro ejemplo es:

$a_1 = \{00\}$ con $p_1 = ½$

$a_2 = \{01\}$ con $p_2 = ¼$

$a_3 = \{10\}$ con $p_3 = ⅛$

$a_4 = \{11\}$ con $p_4 = ⅛$

Definiciónes:

largo promedio : L = [bit]

tasa de bits : R = [bit/s]

L : Largo promedio en bits de un mensaje o letra, se calcula como su esperanza.

R : Depende de la cantidad de mensajes o letras que se quieran transmitir por segundo. Suele ser un valor restrictivo que depende del canal (se retoma más adelante).

Códigos de largo variable:

Otro ejemplo es:

$a_1 = \{00\}$ con $p_1 = ½$

$a_2 = \{01\}$ con $p_2 = ¼$

$a_3 = \{10\}$ con $p_3 = ⅛$

$a_4 = \{11\}$ con $p_4 = ⅛$

$L = ½\,(2 \text{ bit}) + ¼\,(2 \text{ bit}) + ⅛\,(2 \text{ bit}) + ⅛\,(2 \text{ bit}) = 2 \text{ bit}$

Necesito 2 bits por cada letra que quiero transmitir.

Si tengo un mensaje $M = \{X_1, X_2, X_3, \ldots, X_L\}$

Voy a necesitar 2 L bit para transmitirlo.

Codificación de fuente

Códigos

Códigos de largo variable:

Otro ejemplo es:

a_1 = {00} con p_1 = ½

a_2 = {01} con p_2 = ¼

a_3 = {10} con p_3 = ⅛

a_4 = {11} con p_4 = ⅛

Letter	$P[a_k]$	Code I	Code II	Code III
a_1	$\frac{1}{2}$	1	0	0
a_2	$\frac{1}{4}$	00	10	01
a_3	$\frac{1}{8}$	01	110	011
a_4	$\frac{1}{8}$	10	111	111

"*Digital Communication*", John G. Proakis, Masoud Salehi, 5th edition.

<u>Largo promedio</u>:

L_0 = ½ (2 bit) + ¼ (2 bit) + ⅛ (2 bit) + ⅛ (2 bit) = 2 bit

L_I = ½ (1 bit) + ¼ (2 bit) + ⅛ (2 bit) + ⅛ (2 bit) = 1,5 bit

L_{II} = ½ (1 bit) + ¼ (2 bit) + ⅛ (3 bit) + ⅛ (3 bit) = 1.75 bit

L_{III} = ½ (1 bit) + ¼ (2 bit) + ⅛ (3 bit) + ⅛ (3 bit) = 1.75 bit

Códigos de largo variable:

Otro ejemplo es:

a_1 = {00} con p_1 = ½

a_2 = {01} con p_2 = ¼

a_3 = {10} con p_3 = ⅛

a_4 = {11} con p_4 = ⅛

Letter	$P[a_k]$	Code I	Code II	Code III
a_1	$\frac{1}{2}$	1	0	0
a_2	$\frac{1}{4}$	00	10	01
a_3	$\frac{1}{8}$	01	110	011
a_4	$\frac{1}{8}$	10	111	111

"*Digital Communication*", John G. Proakis, Masoud Salehi, 5th edition.

<u>Códigos de decodificación única e instantáneos</u>: Supongamos S={001001…}

C_0 : Si,Si → M={a_1,a_3,a_2}

C_I : No,No → M={a_2,a_1,a_2,a_1} o M={a_2,a_4,a_3}

C_{II} : Si,Si → M={a_1,a_1,a_2,a_1,...}, el 0 indica el fin de la letra o cuando se llega a 3 bits

C_{III} : Si,No → M={a_1,a_2,a_1,a_2}

Códigos de largo variable:

Otro ejemplo es:

a_1 = {00} con p_1 = ½

a_2 = {01} con p_2 = ¼

a_3 = {10} con p_3 = ⅛

a_4 = {11} con p_4 = ⅛

Letter	$P[a_k]$	Code I	Code II	Code III
a_1	$\frac{1}{2}$	1	0	0
a_2	$\frac{1}{4}$	00	10	01
a_3	$\frac{1}{8}$	01	110	011
a_4	$\frac{1}{8}$	10	111	111

"*Digital Communication*", John G. Proakis, Masoud Salehi, 5th edition.

<u>Códigos de decodificación única e instantáneos</u>: Supongamos S={001001…}

C_0 : Si,Si → M={a_1,a_3,a_2}

C_I : No,No → M={a_2,a_1,a_2,a_1} o M={a_2,a_4,a_3}

C_{II} : Si,Si → M={a_1,a_1,a_2,a_1,...}, el 0 indica el

C_{III} : Si,No → M={a_1,a_2,a_1,a_2}

Condición suficiente para ser de decodificación única e instantáneo:
Codewords libres de prefijo

Códigos de largo variable:

Resumen de propiedades

Propiedad	Código 0	Código I	Código II	Código III
largo promedio	2 bit	1,5 bit	1,75 bit	1,75 bit
decod. única	Si	No	Si	Si
instantáneo	Si	No	Si	No

Códigos de largo variable:

Resumen de propiedades

Propiedad	**Código 0**	**Código I**	**Código II**	**Código III**
largo promedio	2 bit	1,5 bit	1,75 bit	1,75 bit
decod. única	Si	No	Si	Si
instantáneo	Si	No	Si	No

Solo queremos códigos de **decodificación única** e **instantáneos** y lo más **cortos** posible.

¿Hay algo mejor que el código II?

Teorema de la codificación de fuente:

SHANNON'S FIRST THEOREM (LOSSLESS SOURCE CODING THEOREM) Let X denote a DMS with entropy X. There exists a lossless source code for this source at any rate R if $R > H(X)$. There exists no lossless code for this source at rates less than $H(X)$.

Teorema de la codificación de fuente:

Para una fuente discreta sin memoria, es posible construir un código que cumpla con:

$$H(X) \leq L \leq H(X) + 1$$

En el ejemplo anterior:

$$H(X) = -(\tfrac{1}{2} \log(\tfrac{1}{2}) + \tfrac{1}{4} \log(\tfrac{1}{4}) + \tfrac{1}{8} \log(\tfrac{1}{8}) + \tfrac{1}{8} \log(\tfrac{1}{8})) = 1{,}75 \text{ bit}$$

Entonces el código 2 alcanza el mejor valor posible para el largo promedio.

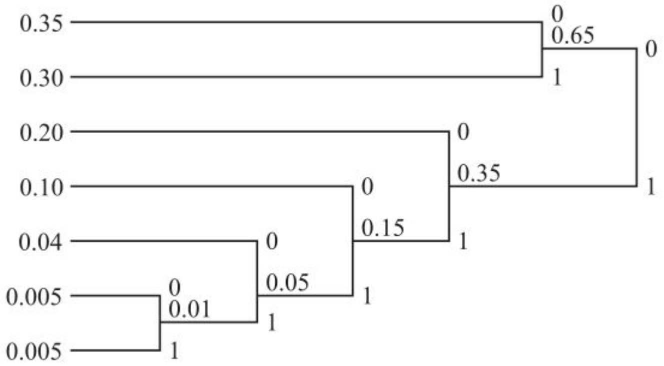
<u>Definiciones</u>:

Eficiencia : $\eta = H(X)/L$ = 1 (para el ejemplo anterior)

Redundancia : $\gamma = 1 - \eta$ = 0 (para el ejemplo anterior)

Codificación de Huffman:

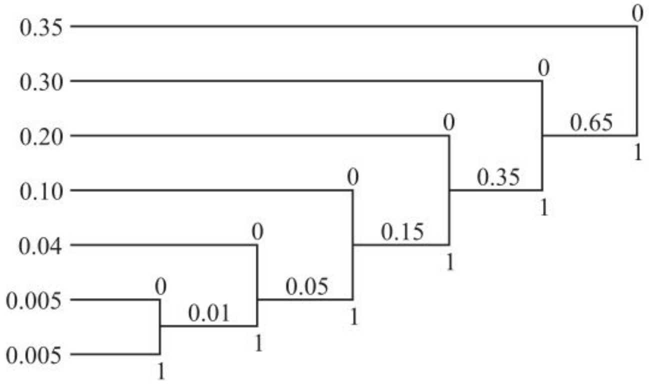
Procedimiento: Supongamos que tenemos los símbolos x_1 a x_7, se construye el código agrupando de a 2 los 2 símbolos menos probables:



Letter	Probability	Self-information	Code
x_1	0.35	1.5146	00
x_2	0.30	1.7370	01
x_3	0.20	2.3219	10
x_4	0.10	3.3219	110
x_5	0.04	4.6439	1110
x_6	0.005	7.6439	11110
x_7	0.005	7.6439	11111
	$H(X) = 2.11$	$\bar{R} = 2.21$	

Codificación de Huffman:

La codificación no es única:

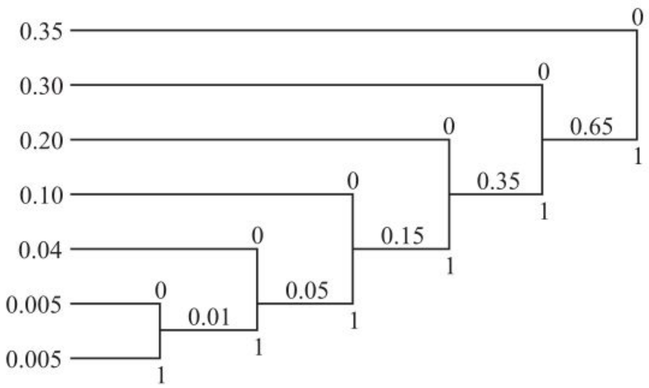


Letter	Code
x_1	0
x_2	10
x_3	110
x_4	1110
x_5	11110
x_6	111110
x_7	111111

$\bar{R} = 2.21$

Codificación de Huffman:

La codificación no es única:



Letter	Code	Code
x_1	0	00
x_2	10	01
x_3	110	10
x_4	1110	110
x_5	11110	1110
x_6	111110	11110
x_7	111111	11111
	$\bar{R} = 2.21$	$\bar{R} = 2.21$

H(X) = 2,11

L = 2,21

η = H(X)/L = 0.95475

Codificación de Huffman:

Alcanza la eficiencia de manera asintótica, es decir codificando por bloques de símbolos o letras.

$N \rightarrow \infty$

$L \rightarrow H(X)$

$\eta = H(X)/L \rightarrow 1$

$\gamma \rightarrow 0$

Letter	Probability	Self-information	Code
x_1	0.45	1.156	1
x_2	0.35	1.520	00
x_3	0.20	2.330	01

$H(X) = 1.513$ bits/letter
$\bar{R}_1 = 1.55$ bits/letter
Efficiency $= 97.6\%$

Letter pair	Probability	Self-information	Code
x_1x_1	0.2025	2.312	10
x_1x_2	0.1575	2.676	001
x_2x_1	0.1575	2.676	010
x_2x_2	0.1225	3.039	011
x_1x_3	0.09	3.486	111
x_3x_1	0.09	3.486	0000
x_2x_3	0.07	3.850	0001
x_3x_2	0.07	3.850	1100
x_3x_3	0.04	4.660	1101

$2H(X) = 3.026$ bits/letter pair
$\bar{R}_2 = 3.0675$ bits/letter pair
$\frac{1}{2}\bar{R}_2 = 1.534$ bits/letter
Efficiency $= 98.6\%$

Codificación Lempel-Ziv:

Empecemos viendo una mala propiedad de la codificación de Huffman:

> Para hallar el código es necesario conocer a la perfección las probabilidades de cada símbolo y de cada conjunto de símbolos.

Esto es impráctico en la mayoría de las ocasiones.

La codificación de Lempel-Ziv es una **codificación de fuentes universales**, es decir que sirve para todo tipo de fuente, independientemente de sus propiedades.

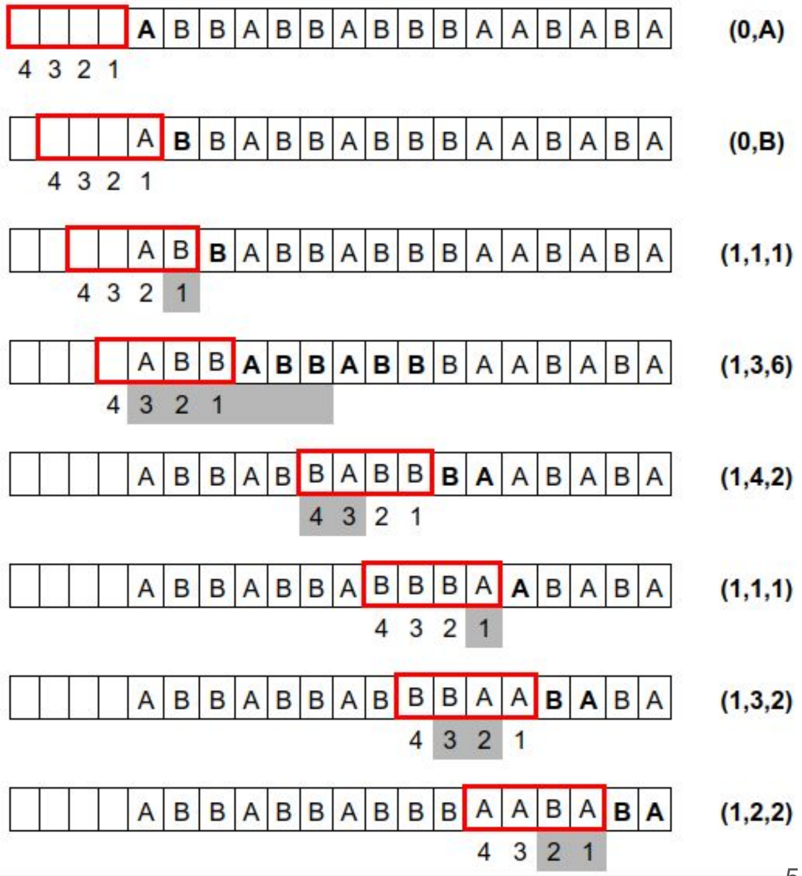
Es una codificación basada en **diccionario adaptativo**.

Hay dos grupos: ”**ventana deslizante**” y “**estructurado en forma de árbol**”.

Codificación Lempel-Ziv:

Ejemplo de codificación Lempel-Ziv del tipo “ventana deslizante”.

- Se buscan cadenas repetidas en una ventana.
- Se codifica con (F,C) o (F,P,L)
  - F : Flag, indicando ocurrencia
  - C : Caracter
  - P : Posición (contada desde atrás)
  - L : Largo
- Se utiliza en gzip y pkzip.
- Una variación de éste es óptimo asintóticamente.



Codificación Lempel-Ziv:

Ejemplo de codificación Lempel-Ziv del tipo "estructurada en forma de árbol":

- Códigos de largo fijo
- Se arma un diccionario dinámicamente
- [posición] + [último bit]
- La tabla puede crecer indefinidamente
    - Se limita el largo
    - Se utiliza una política de reemplazo
- Se utiliza en el formato gif
- Es asintóticamente óptimo

10101101001001110101000011001110101100011011

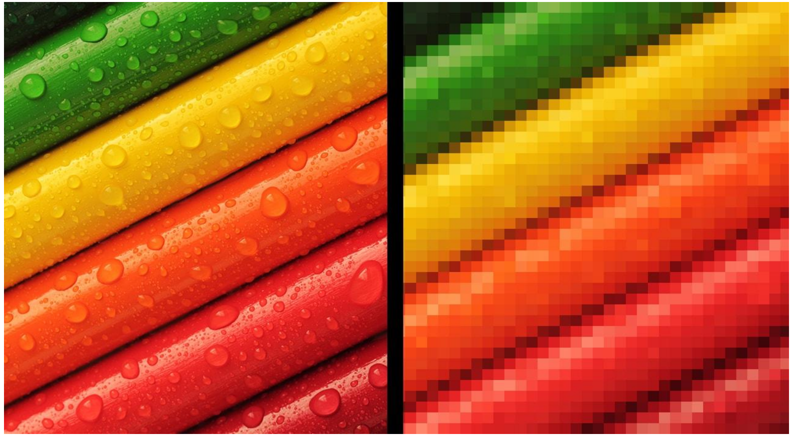
1, 0, 10, 11, 01, 00, 100, 111, 010, 1000, 011, 001, 110, 101, 10001, 1011

	Dictionary location	Dictionary contents	Code word
1	0001	1	00001
2	0010	0	00000
3	0011	10	00010
4	0100	11	00011
5	0101	01	00101
6	0110	00	00100
7	0111	100	00110
8	1000	111	01001
9	1001	010	01010
10	1010	1000	01110
11	1011	011	01011
12	1100	001	01101
13	1101	110	01000
14	1110	101	00111
15	1111	10001	10101
16		1011	11101

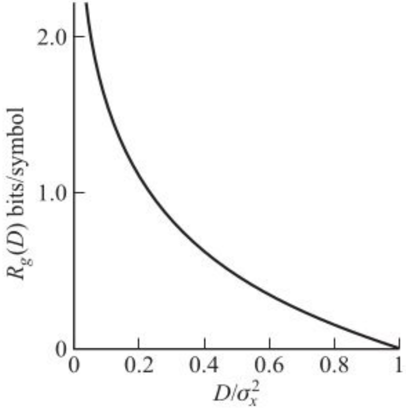
Codificación con pérdida:

Los ejemplos anteriores son capaces de recuperar el mensaje original.

Hay otros códigos que reducen aún más el largo de los datos, pero no son capaces de recuperar el mensaje original, aunque sí sus partes más "importantes".



La performance de estos algoritmos se mide con la curva de "*Rate-Distortion*".



Codificación de fuente

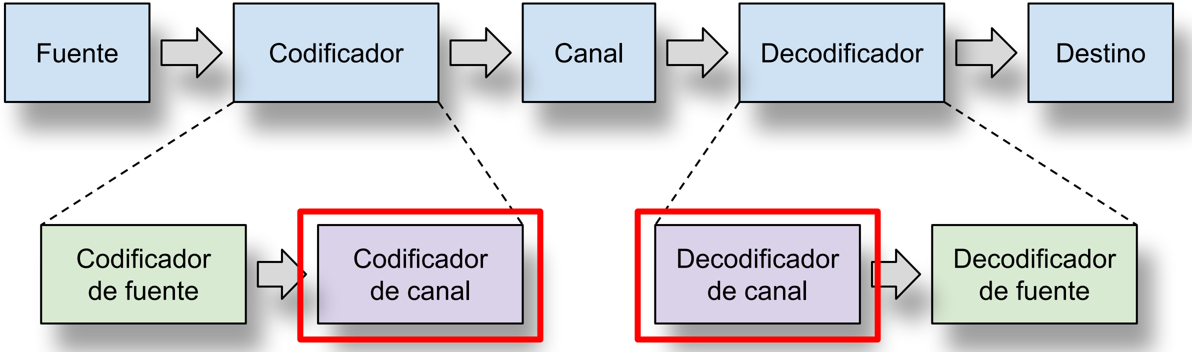
Resumen

Características deseables de los códigos de fuente:

- Decodificable de manera **unívoca**, **instantánea** (o que cumpla con la **condición de prefijo**)
- Obtener **secuencia de símbolos sin redundancia**:
    - Una secuencia de símbolos de largo mínimo
    - Una “nueva fuente” con máxima entropía
    - Una fuente que no se puede comprimir
    - Eficiencia η=1
- Que codifique **fuentes universales**:
    - Que no sea necesario conocer previamente las estadísticas de las fuentes

Codificación de canal:

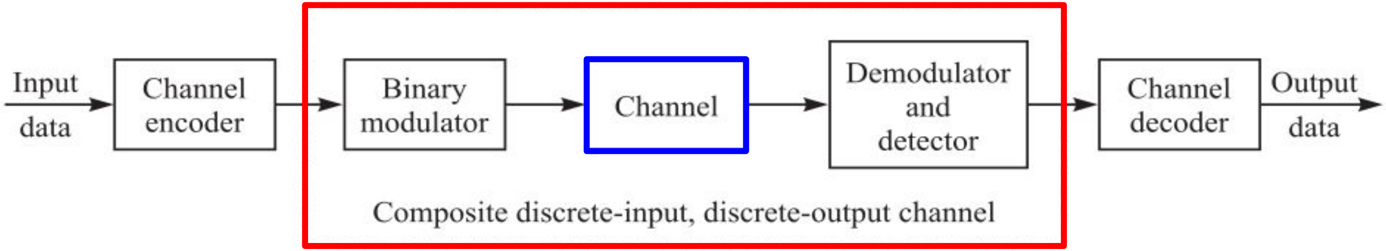
- **Canal**
  - **Modelos**
  - **Capacidad**
- **Codificación**
  - **Detección de errores**
  - **Corrección de errores**
  - **Cod. por bloque**
  - **Cod. convolucional**
  - **Algoritmo Viterbi**
  - **Entrelazado**
  - **Cod. de línea**



El objetivo principal es revertir los efectos distorsivos del canal para obtener transmisiones **libres de errores**.

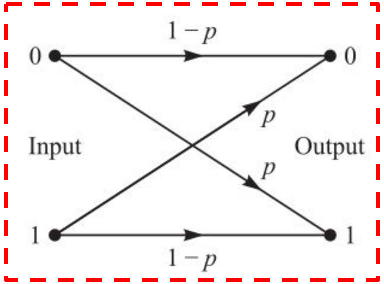
Modelos de canal

Fuente: "*Digital Communication*", John G. Proakis, Masoud Salehi, 5th edition.

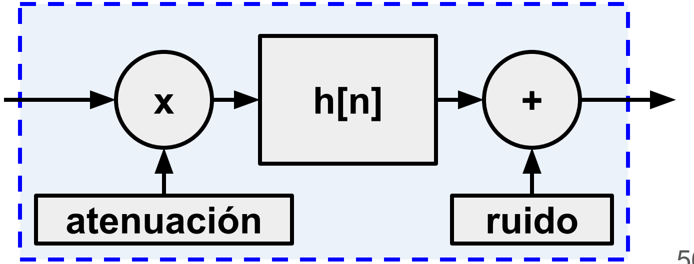


Desde el punto de vista de la **teoría de información**:

- Canal de **entradas y salidas discretas**
- Recibe y devuelve **símbolos**
- Se caracterizan a través de ***P(Y | X)***.

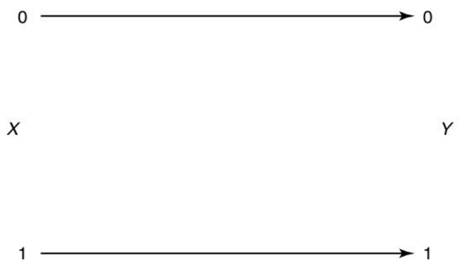


Nota: Es distinto al canal y los modelos utilizados en la **capa física** de los sistemas de comunicaciones, donde el canal es **continuo**.

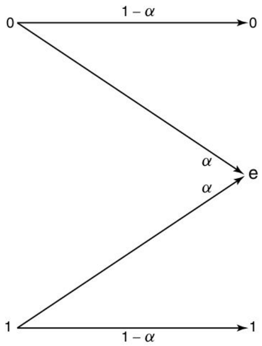


Modelos de canal

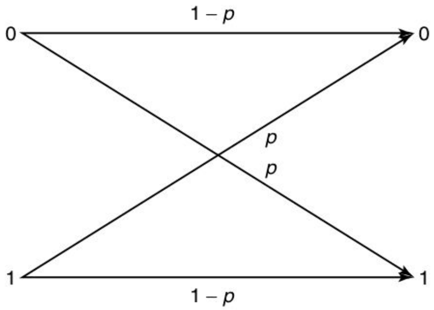
Canal binario sin ruido



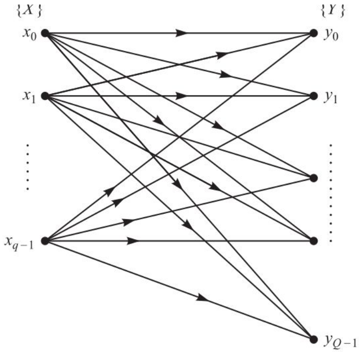
Canal binario + detección de error



Canal binario simétrico (BSC)



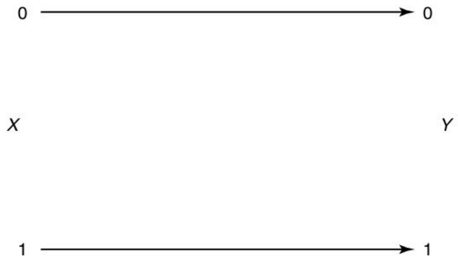
Canal genérico sin memoria



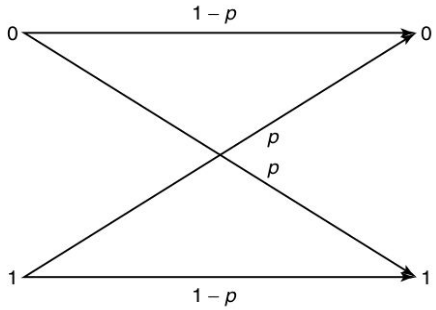
Codificación de canal

Modelos de canal

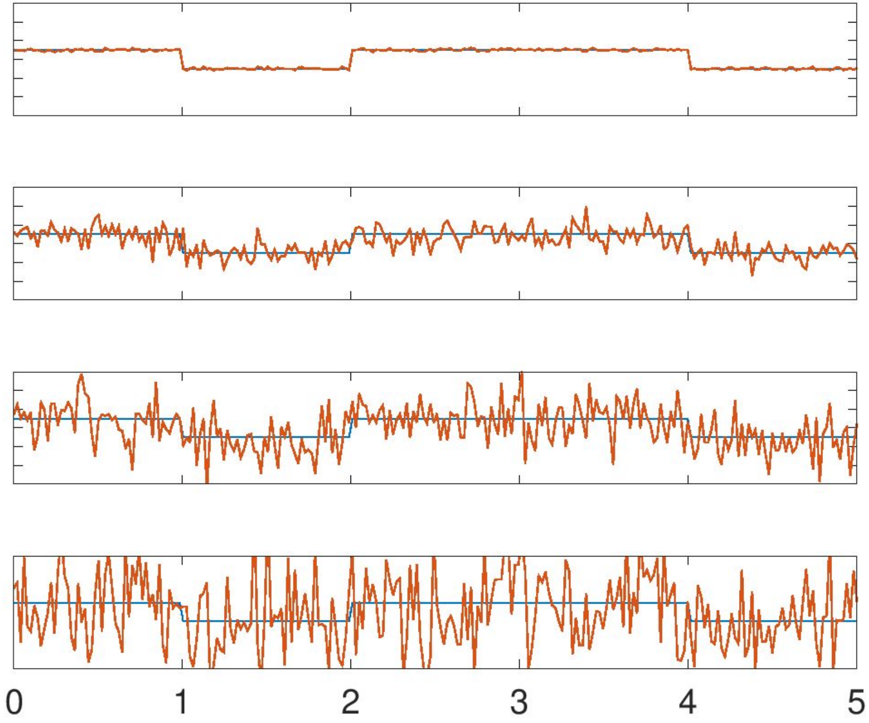
Canal binario sin ruido



Canal binario simétrico (BSC)



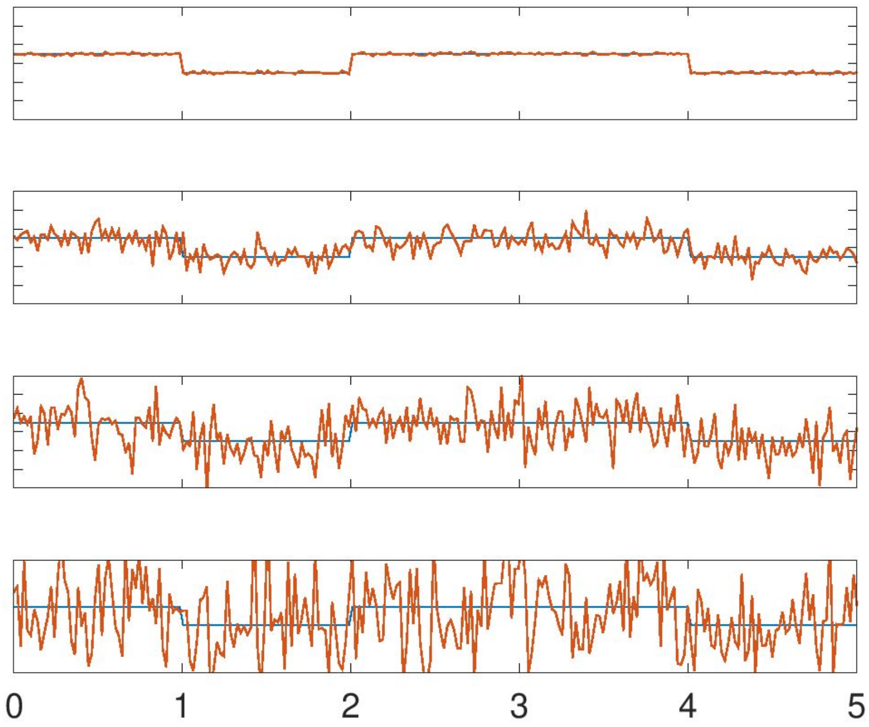
Canal continuo en la capa física



¿Es posible enviar mensajes **sin errores** a través de un **canal con ruido**?

Durante mucho tiempo, se creyó que esto no era posible, que solo era posible reducir la probabilidad de error aumentando la relación que hay entre la señal y el ruido.

Canal continuo en la capa física

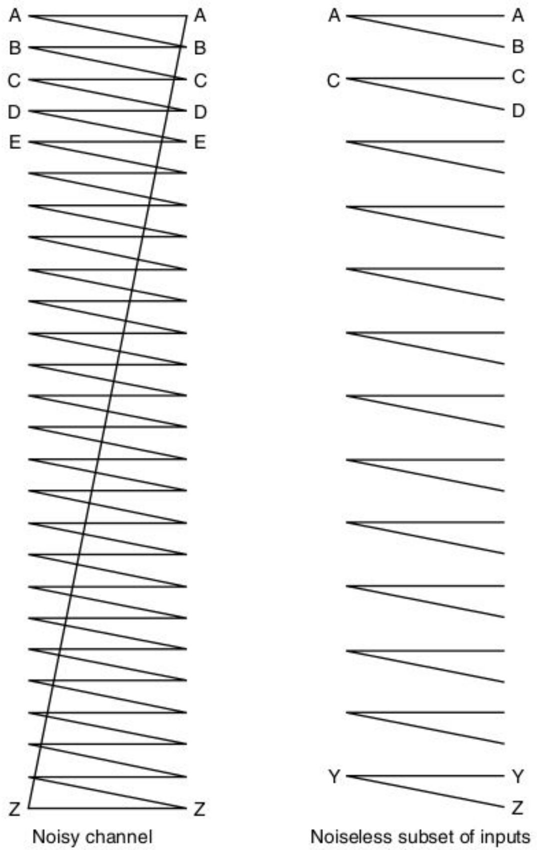


Modelos de canal

¿Es posible enviar mensajes **sin errores** a través de un **canal con ruido**?

Durante mucho tiempo, se creyó que esto no era posible, que solo era posible reducir la probabilidad de error aumentando la relación que hay entre la señal y el ruido.

Se demuestra que sí es posible.



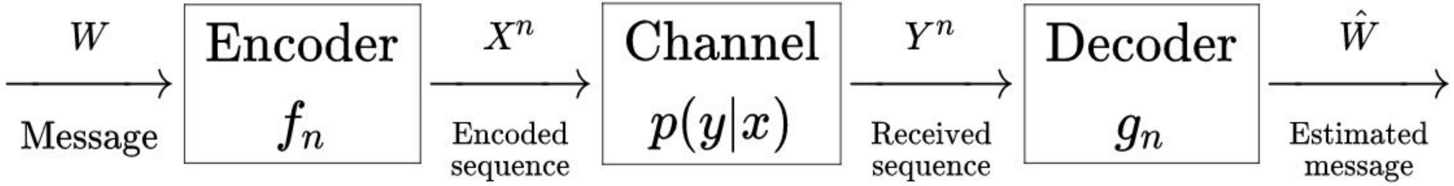
Codificación de canal

Capacidad del canal

Capacidad del canal: Concepto introducido por Shannon:

> **SHANNON'S SECOND THEOREM—THE NOISY CHANNEL CODING THEOREM (SHANNON 1948)**
> Reliable communication over a discrete memoryless channel is possible if the communication rate R satisfies $R < C$, where C is the channel capacity. At rates higher than capacity, reliable communication is impossible.

Capacidad del canal: Concepto introducido por Shannon:

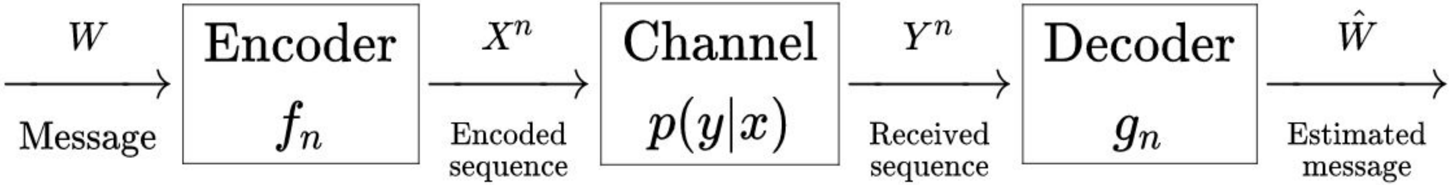


$$C = \sup_{p_X(x)} I(X;Y)$$

where the supremum is taken over all possible choices of $p_X(x)$.

Fuente: https://en.wikipedia.org/wiki/Channel_capacity

Capacidad del canal: Concepto introducido por Shannon:



$$C = \sup_{p_X(x)} I(X;Y)$$

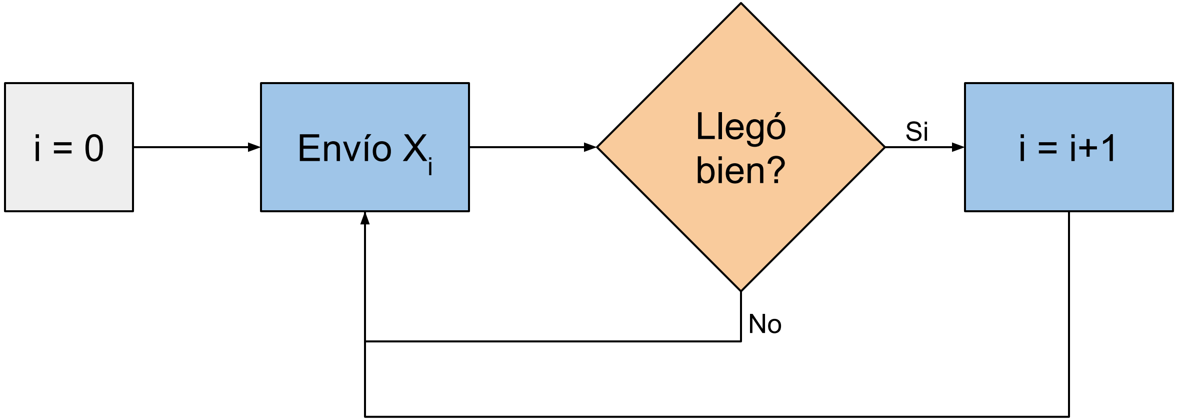
¿?

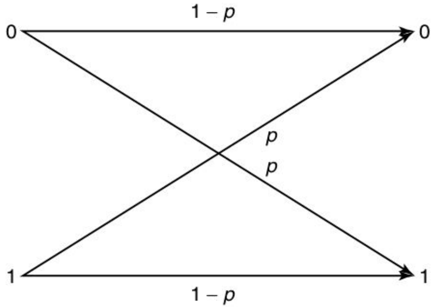
where the supremum is taken over all possible choices of $p_X(x)$.

Fuente: https://en.wikipedia.org/wiki/Channel_capacity

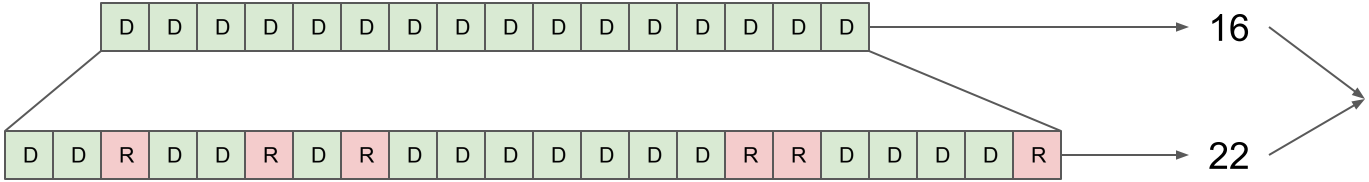
Capacidad del canal

Capacidad del canal: Un ejemplo intuitivo.





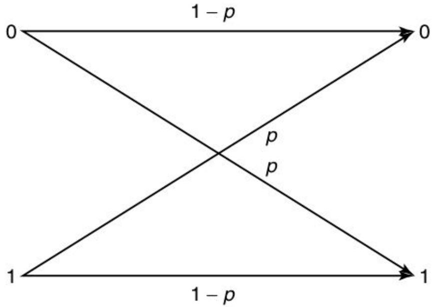
¿Cuántas veces tengo que mandar un dato en promedio hasta que llegue bien?



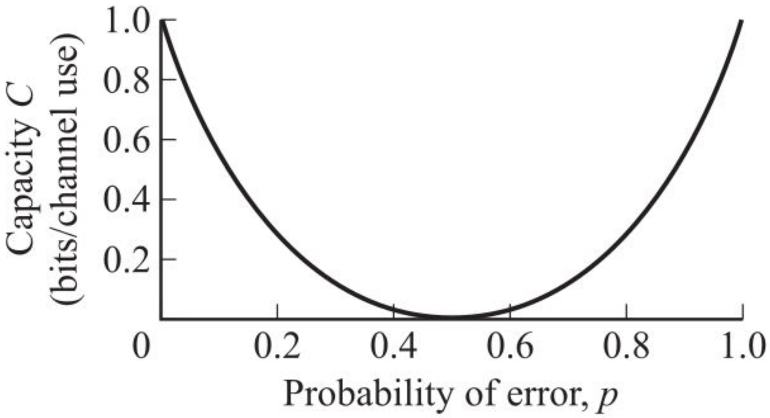
C = 16/22 = 0.727…

Capacidad del canal: Ejemplo concreto.

Capacidad del canal para el canal BSC (discreto):



$$C = 1 + p \log 2p + (1 - p) \log 2(1 - p) = 1 - H(p)$$



Unidad de medida:
[bits/uso del canal]

Capacidad del canal

Capacidad del canal para el canal AWGN (contínuo):

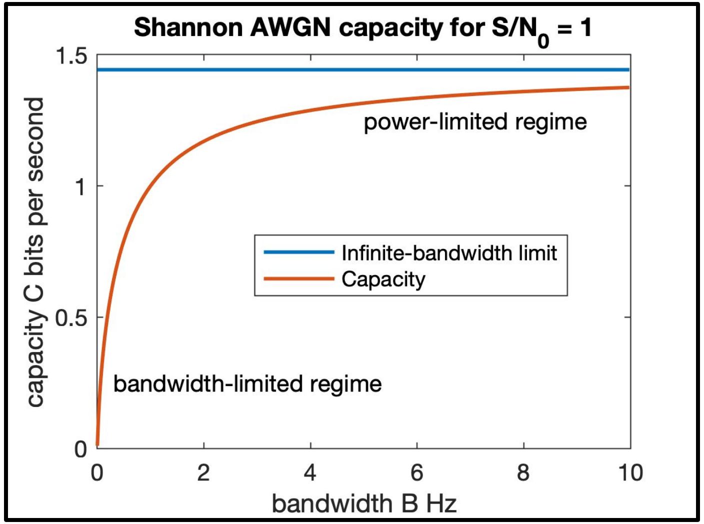
Unidad de medida:
[bits/s]

$$\boxed{C = W \log_2\left(1 + \frac{P}{N_0 W}\right) \qquad \text{bits/s}}$$

$$\boxed{\begin{aligned} P &\to \infty \\ C &\to \infty \end{aligned}}$$

$$\boxed{C_\infty = \lim_{W\to\infty} W \log_2\left(1 + \frac{P}{N_0 W}\right) = (\log_2 e)\,\frac{P}{N_0} \approx 1.44\,\frac{P}{N_0} \qquad \text{bits/s}}$$

Capacidad del canal: Algunas relaciones



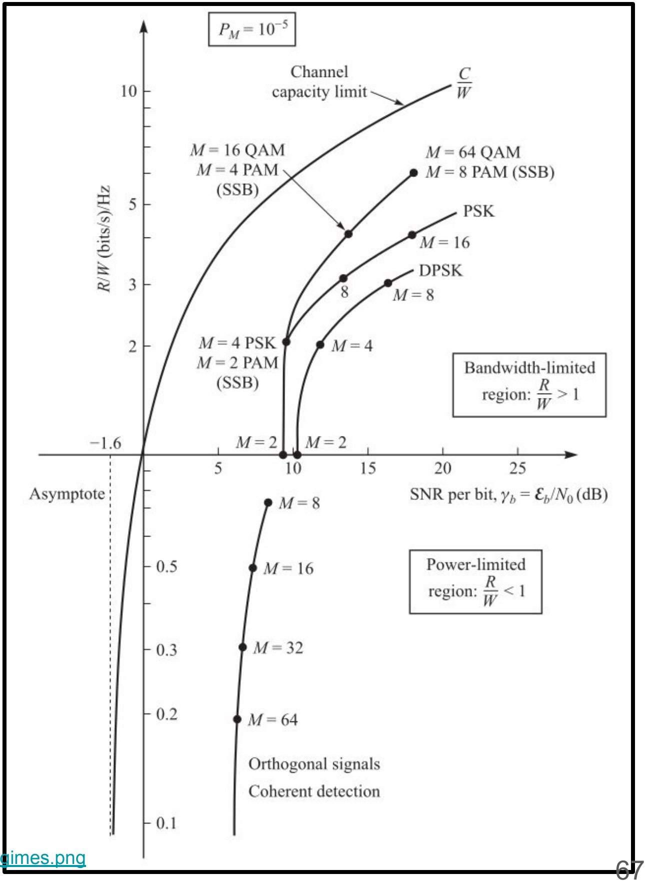
$$R < C$$

$$C_\infty \approx 1.44 \frac{P}{N_0} \quad \text{bits/s}$$

$$P \to \infty$$
$$C \to \infty$$

$$\frac{\mathcal{E}_b}{N_0} > \ln 2 \approx 0.693 \sim -1.6 \text{ dB}$$

$$C = W \log_2 \left(1 + \frac{P}{N_0 W}\right) \quad \text{bits/s}$$



Fuente: https://en.wikipedia.org/wiki/Shannon%E2%80%93Hartley_theorem#/media/File:Channel_Capacity_with_Power-_and_Bandwidth-Limited_Regimes.png

Capacidad del canal

Capacidad del canal:

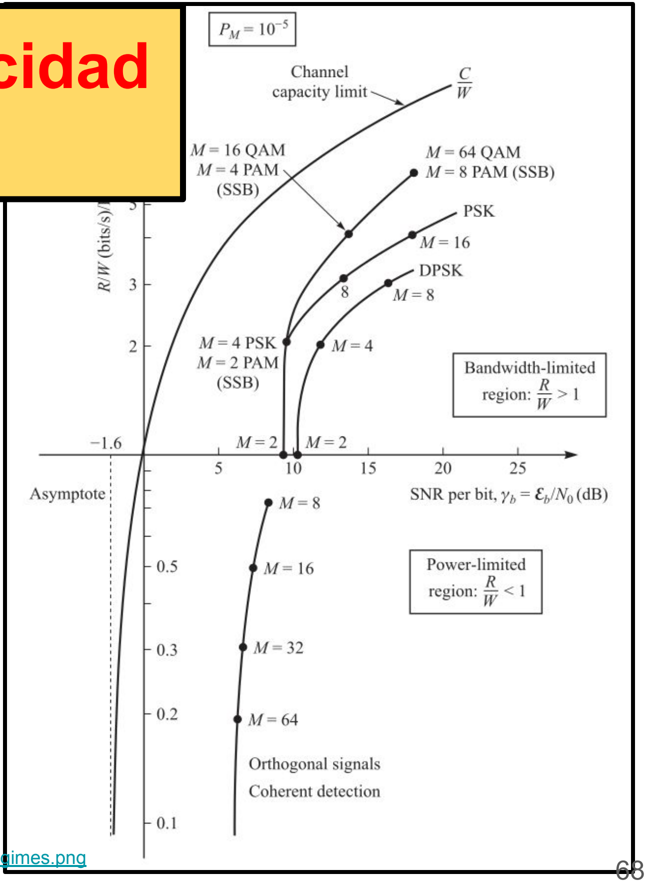
¿Cómo alcanzar la capacidad del canal?

$$C = W \log_2\left(1 + \frac{P}{N_0 W}\right) \qquad \text{bits/s}$$

$$C_\infty \approx 1.44 \frac{P}{N_0} \qquad \text{bits/s}$$

$$P \to \infty$$
$$C \to \infty$$

$$\frac{\mathcal{E}_b}{N_0} > \ln 2 \approx 0.693 \sim -1.6 \text{ dB}$$



Fuente: https://en.wikipedia.org/wiki/Shannon%E2%80%93Hartley_theorem#/media/File:Channel_Capacity_with_Power-_and_Bandwidth-Limited_Regimes.png

Códigos de detección de error:

Paridad:

- Detecta errores impares
- Simple

7 bits of data	(count of 1-bits)	8 bits including parity	
		even	odd
0000000	0	00000000	10000000
1010001	3	11010001	01010001
1101001	4	01101001	11101001
1111111	7	11111111	01111111

Fuente: https://en.wikipedia.org/wiki/Parity_bit

```
11010011101100 000
1101
01100011101100 000
 1101
00111011101100 000
  1101
00010111101100 000
   1101
00000001101100 000
       1101
00000000110100 000
        1101
00000000011000 000
         1101
00000000001110 000
          1101
00000000000101 000
           101 1
00000000000000 100
```

Cyclic Redundancy Check (CRC):

- Detecta distintos tipos de errores
- Simple

Agregan **redundancia**, existe una **trade-off** entre el **overhead**, la capacidad de **detección** y **penalización**. Necesitan **realimentación** entre receptor y transmisor para corregir errores.

Códigos de corrección de error:

No solo detectan, sino que son capaces de corregir algunos de los errores que aparecen sin la necesidad de una retransmisión de datos (realimentación entre receptor y transmisor).

Al igual que antes:

- El código agrega **redundancia**
- Existe una **trade-off** entre el **overhead**, la capacidad de **corrección**

Hay dos tipos de códigos que se utilizan:

- Códigos por **bloques lineales**
- Códigos **convolucionales** (lineales)

Caracterizamos un código según:

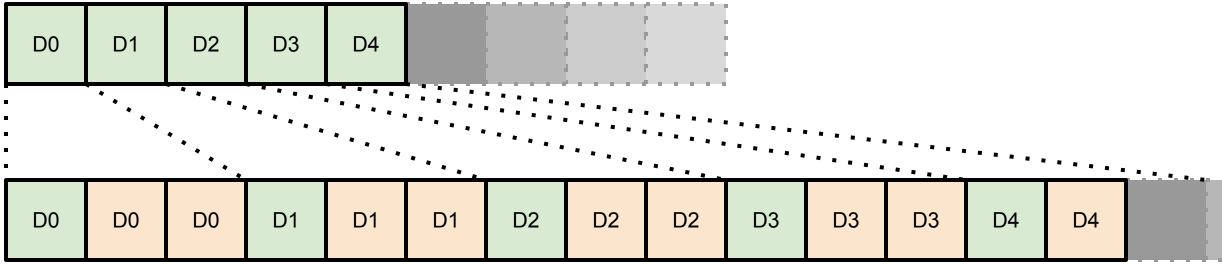
- Su capacidad de corrección
- Su eficiencia o tasa η = k/n
- La complejidad computacional/espacial para la codificación y decodificación

Códigos

Códigos por bloques lineales:

Códigos de repetición:

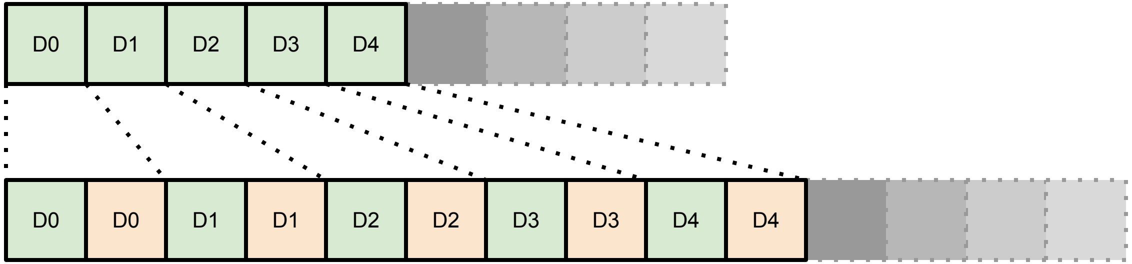
Se repite el dato n veces.



Su tasa es de 1/n.

Si n es impar puede corregir hasta (n-1)/2 errores.

Si n es par puede corregir hasta (n-2)/2 errores.



Códigos por bloques lineales:

Códigos de Hamming:

Se crean generando bits de **paridad** para subconjuntos de los bits de datos.

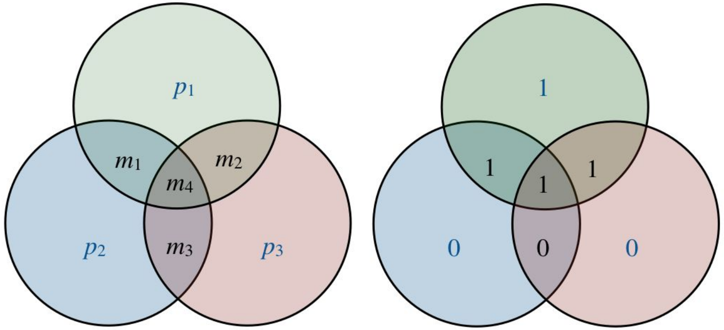
Pueden corregir hasta **1 error**.

Su **tasa** tiende a 1.

Parity bits	Total bits	Data bits	Name	Rate
2	3	1	Hamming(3,1) (Triple repetition code)	1/3 ≈ 0.333
3	7	4	Hamming(7,4)	4/7 ≈ 0.571
4	15	11	Hamming(15,11)	11/15 ≈ 0.733
5	31	26	Hamming(31,26)	26/31 ≈ 0.839
6	63	57	Hamming(63,57)	57/63 ≈ 0.905
7	127	120	Hamming(127,120)	120/127 ≈ 0.945
8	255	247	Hamming(255,247)	247/255 ≈ 0.969
		...		
m	$n = 2^m - 1$	$k = 2^m - m - 1$	Hamming$(2^m - 1, 2^m - m - 1)$	$(2^m - m - 1)/(2^m - 1)$

Fuente: https://en.wikipedia.org/wiki/Hamming_code

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	...	1	1
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	...	1	1
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	...	1	1
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	...	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	...	0	1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18		30	31
P	P	D	P	D	D	D	P	D	D	D	D	D	D	D	P	D	D	...	D	D
p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13		d25	d26



Fuente: https://thatsmaths.com/2015/09/24/hammings-smart-error-correcting-codes/

Códigos por bloques lineales:

Otros códigos conocidos

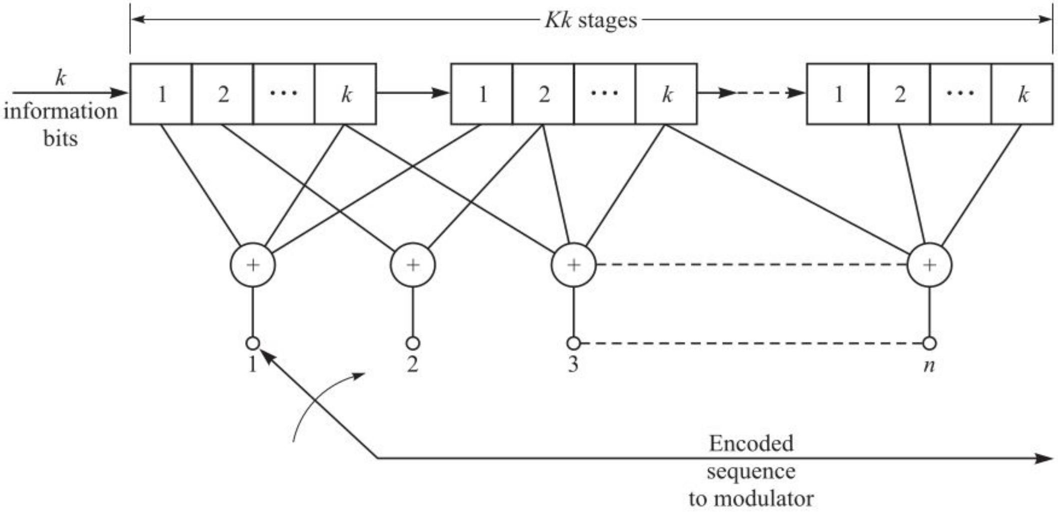
- Códigos Reed-Muller
- Códigos Hardamard
- Códigos Golay
- Códigos Reed-Solomon
- Códigos Low-Density Parity-Check (LDPC)
- Códigos Cíclicos

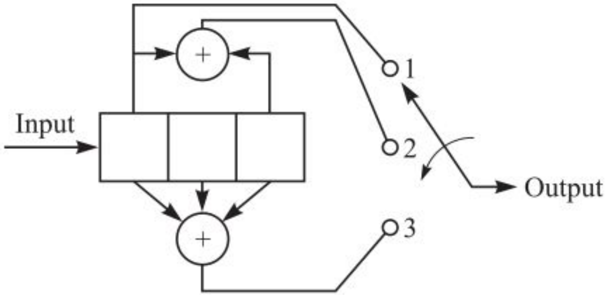
Códigos convolucionales:

K : etapas — K = 3

k : bits de datos — k = 1

n : bits de código — n = 3



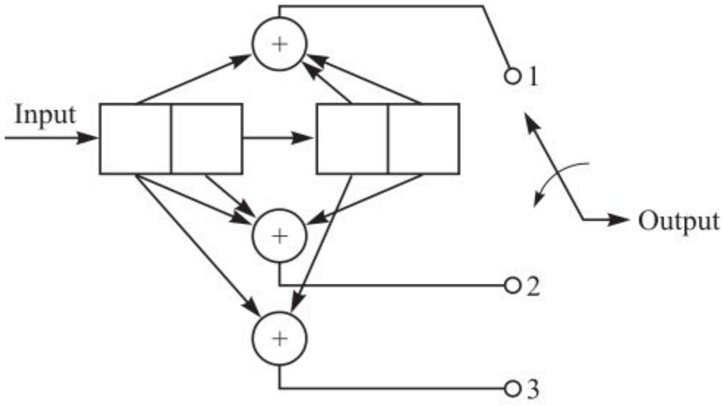


Códigos convolucionales:

K = 2

k = 2

n = 3

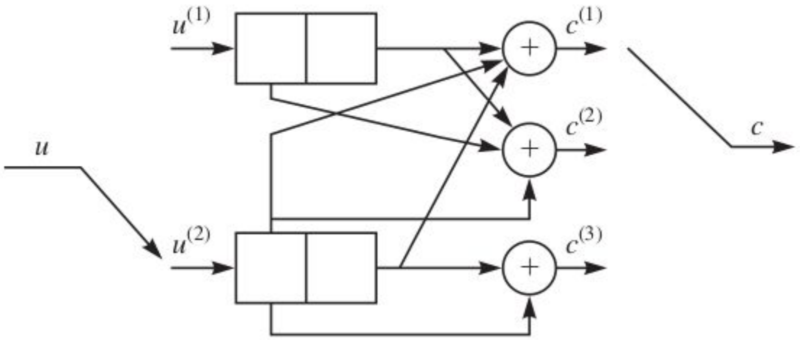


Implementación 1

K = 2

k = 2

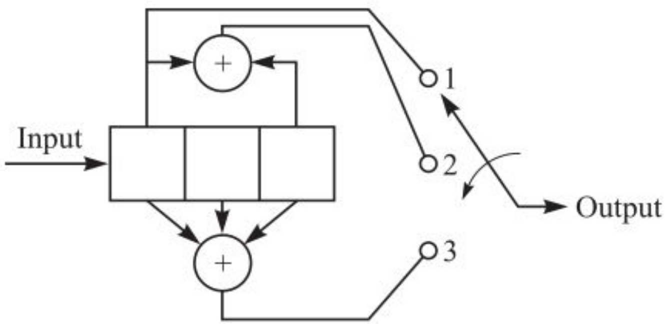
n = 3



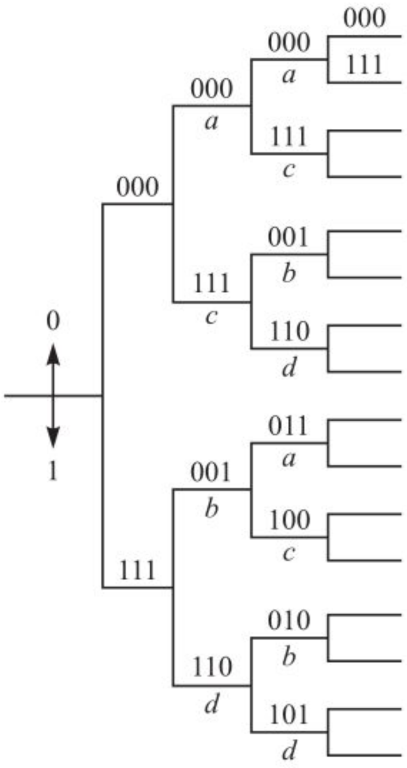
Implementación 2

Códigos convolucionales:

K = 3, k = 1, n = 3

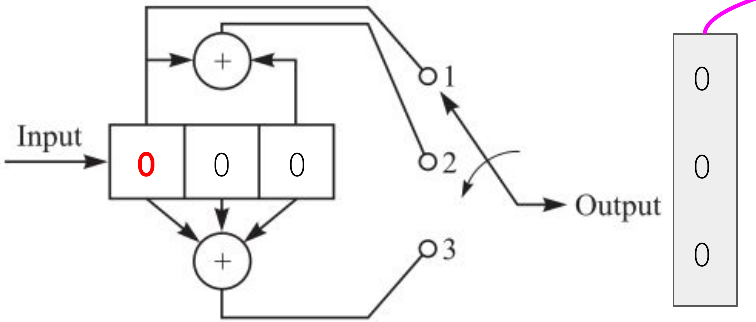


Representación en forma de árbol



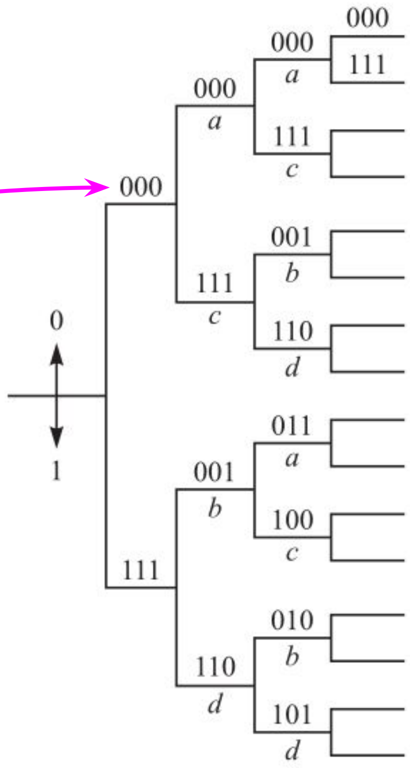
Códigos convolucionales:

K = 3, k = 1, n = 3



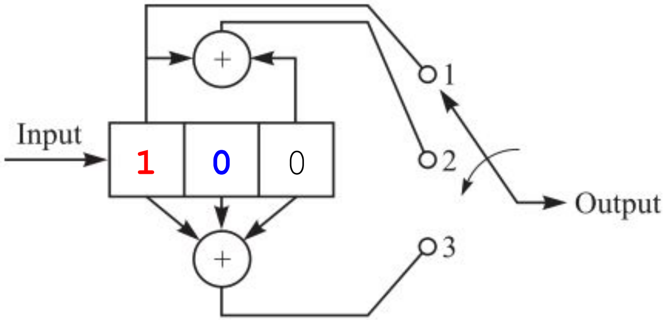
0 1 1 0

Representación en forma de árbol



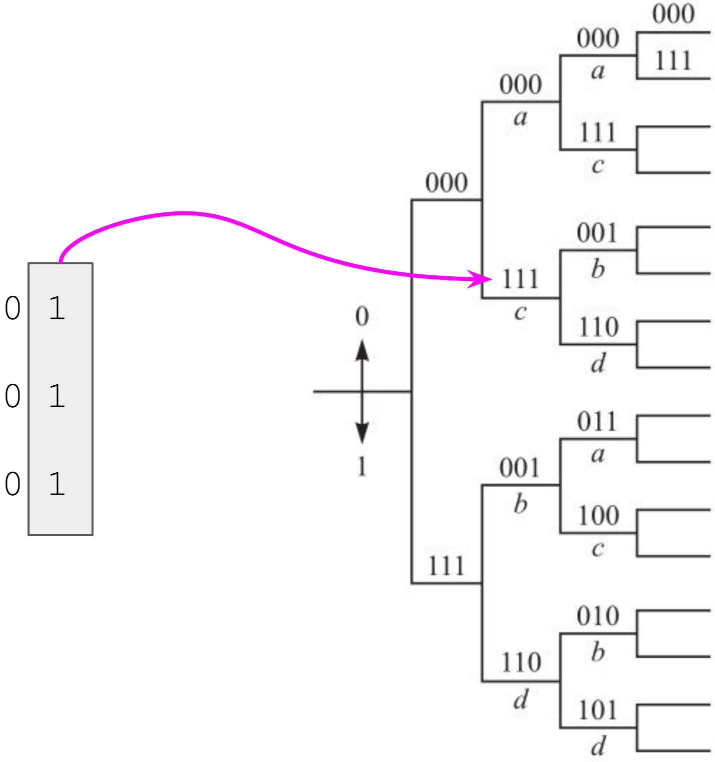
Códigos convolucionales:

K = 3, k = 1, n = 3



0 1 1 0

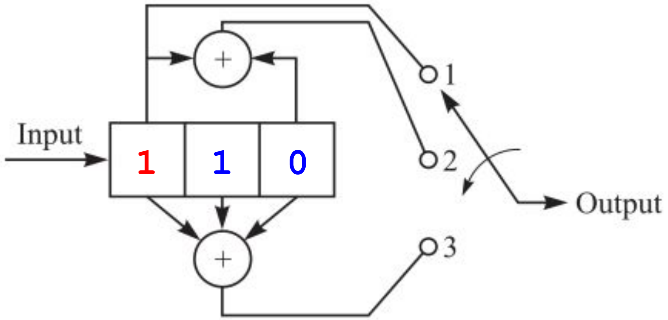
Representación en forma de árbol



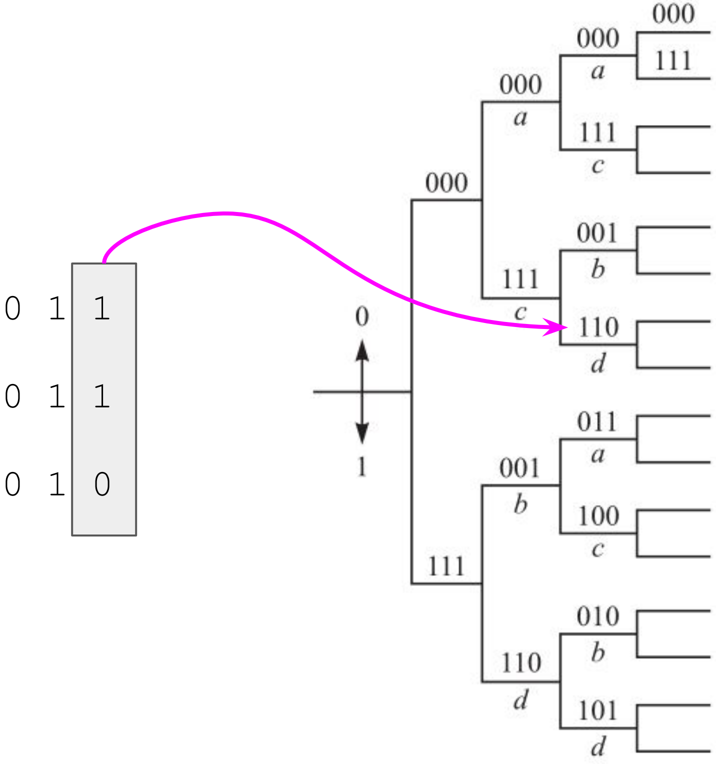
Códigos convolucionales:

Representación en forma de árbol

K = 3, k = 1, n = 3

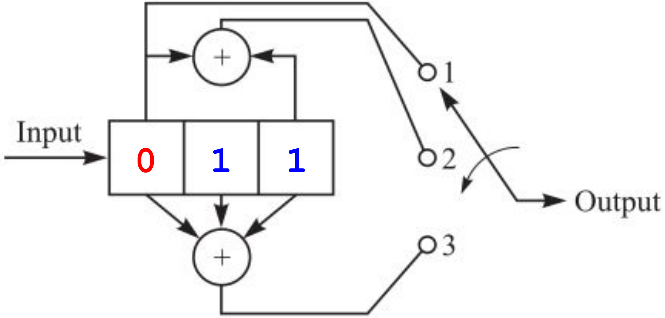


0 1 1 0



Códigos convolucionales:

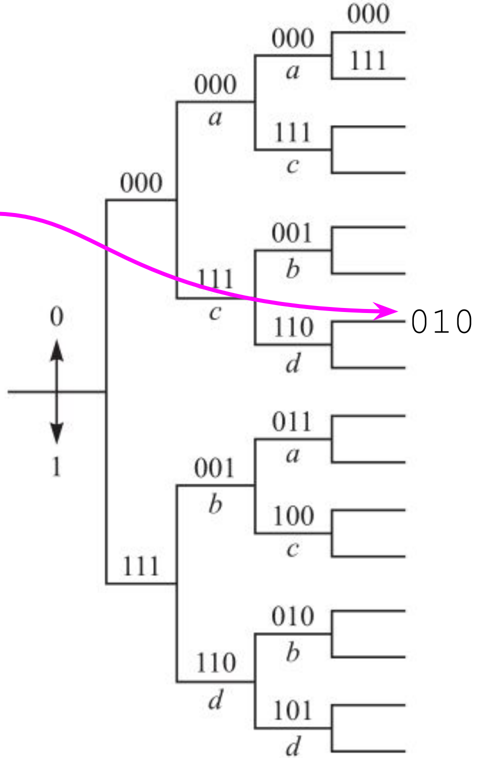
K = 3, k = 1, n = 3



0 1 1 0

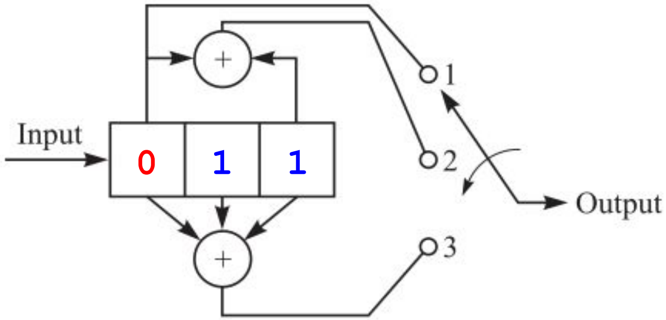
0	1	1	0
0	1	1	1
0	1	0	0

Representación en forma de árbol



Códigos convolucionales:

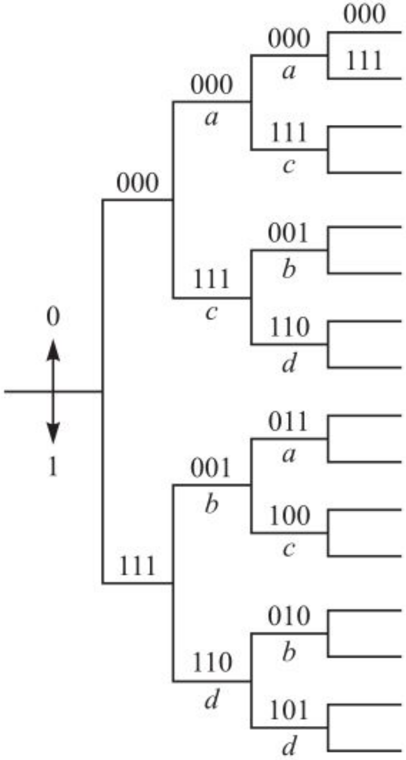
K = 3, k = 1, n = 3



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

0 1 1 0

Representación en forma de árbol



010

000111110010

Códigos convolucionales:

K = 3, k = 1, n = 3

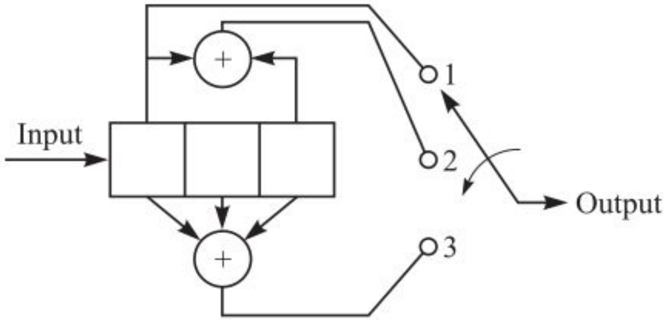
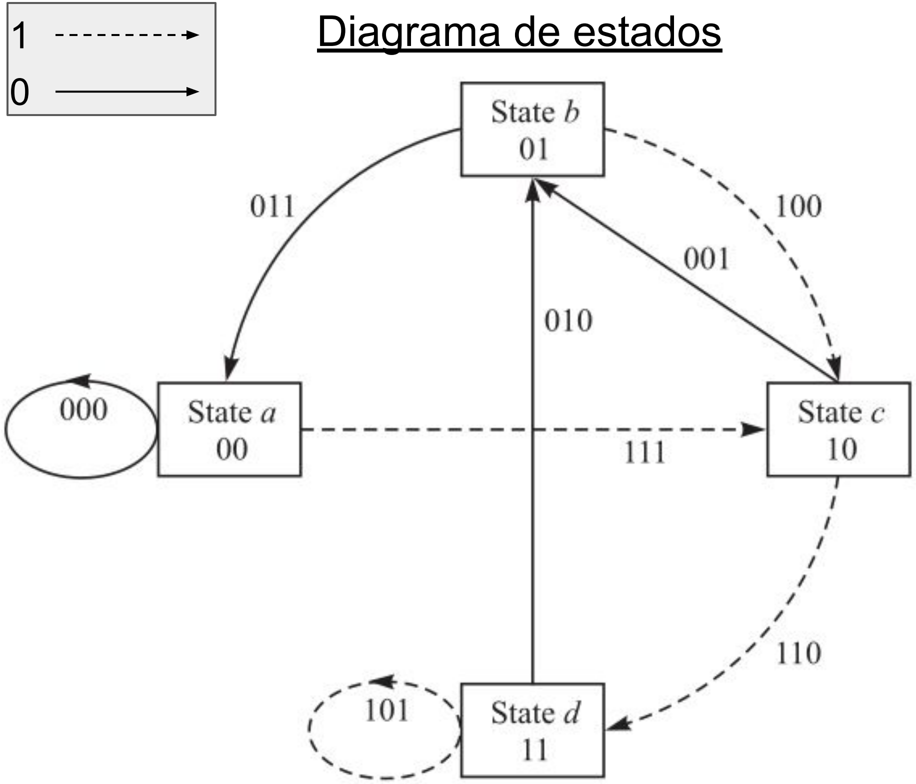
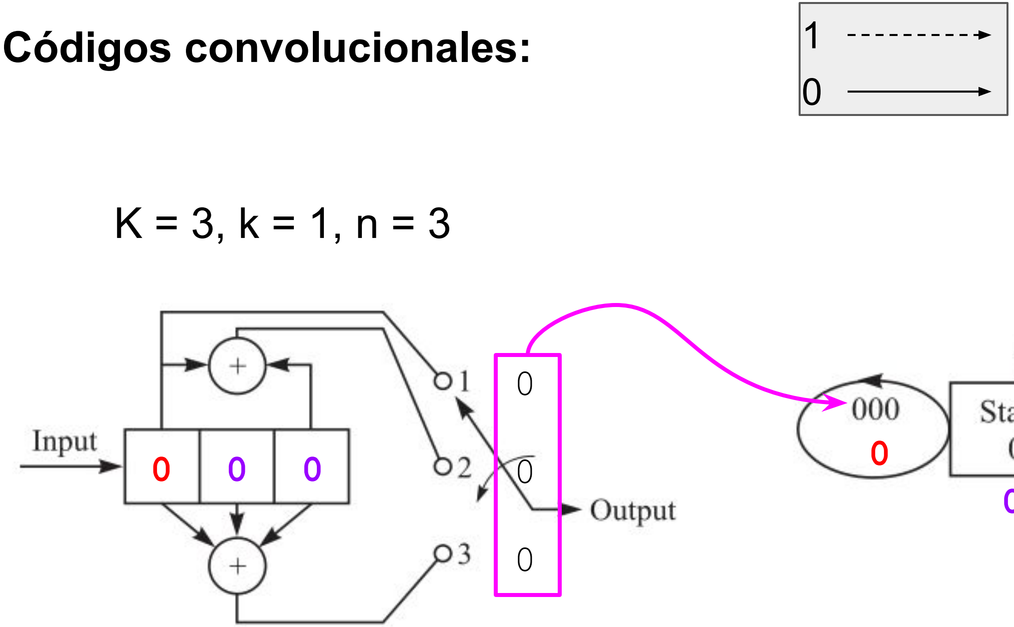


Diagrama de estados

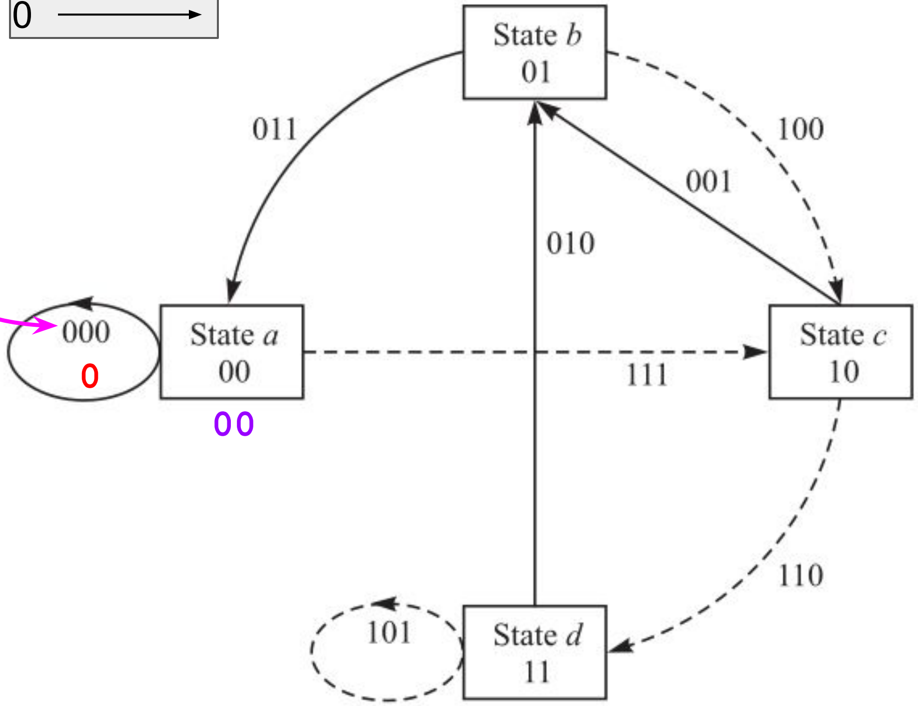


Códigos convolucionales:

$K = 3, k = 1, n = 3$

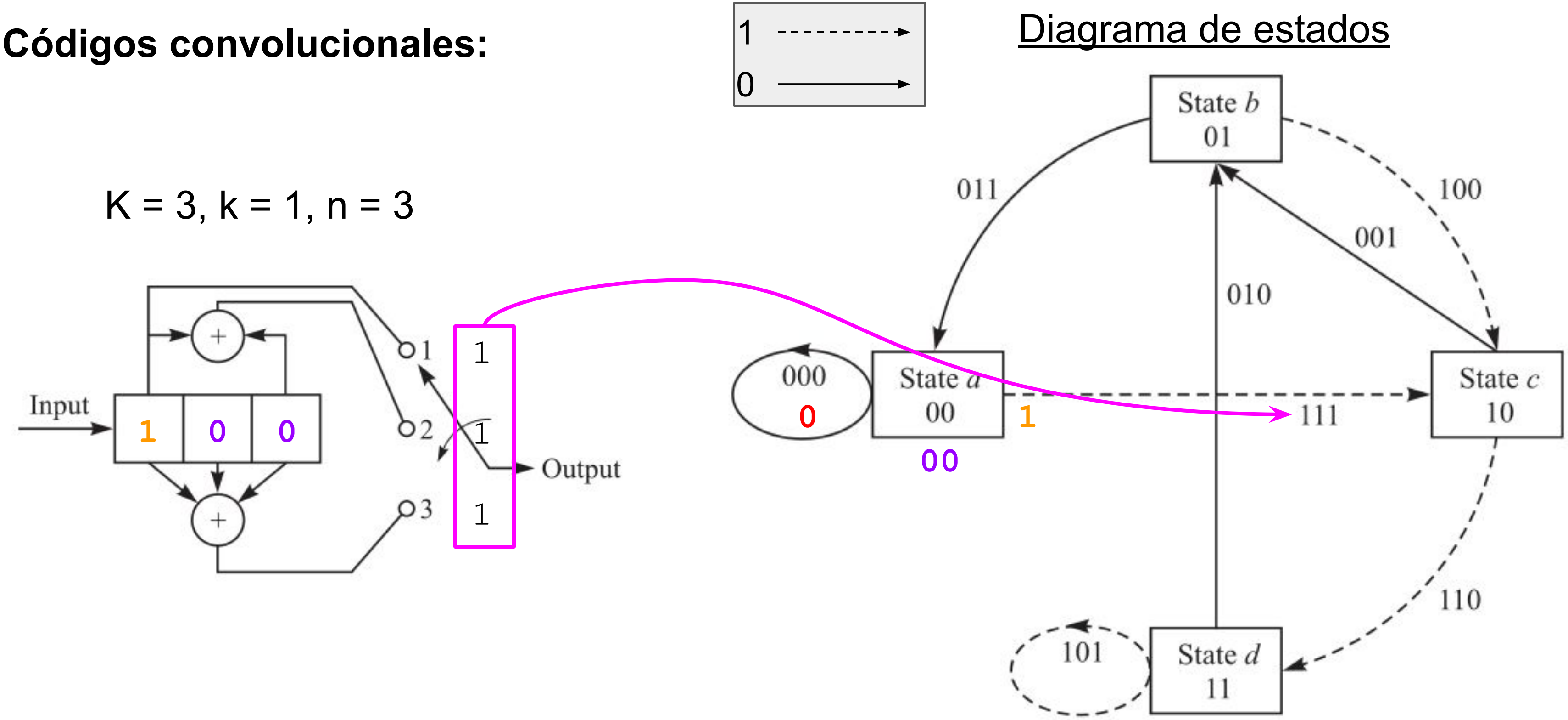


<u>Diagrama de estados</u>



Códigos convolucionales:

K = 3, k = 1, n = 3



Códigos convolucionales:

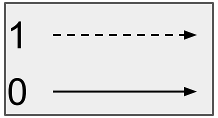
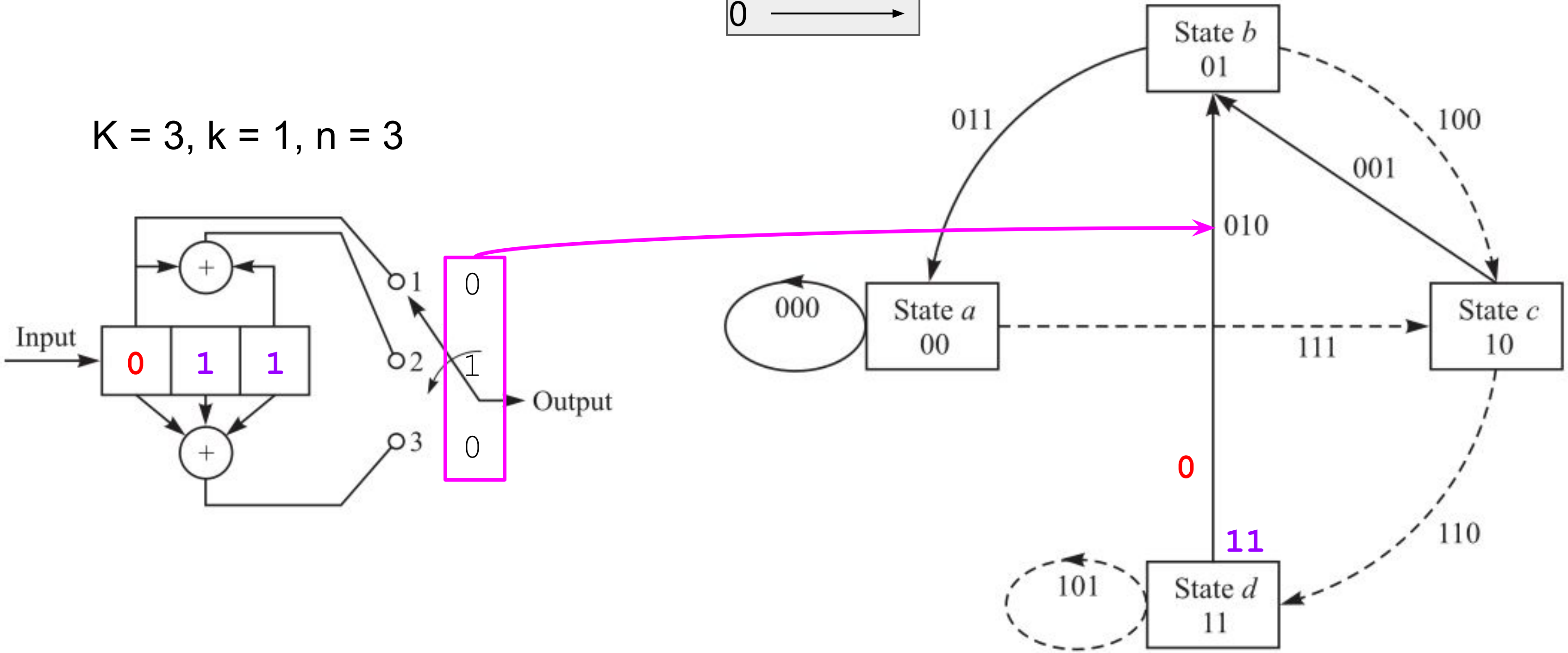


Diagrama de estados

K = 3, k = 1, n = 3



Códigos convolucionales:

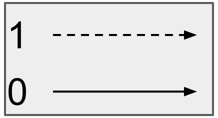
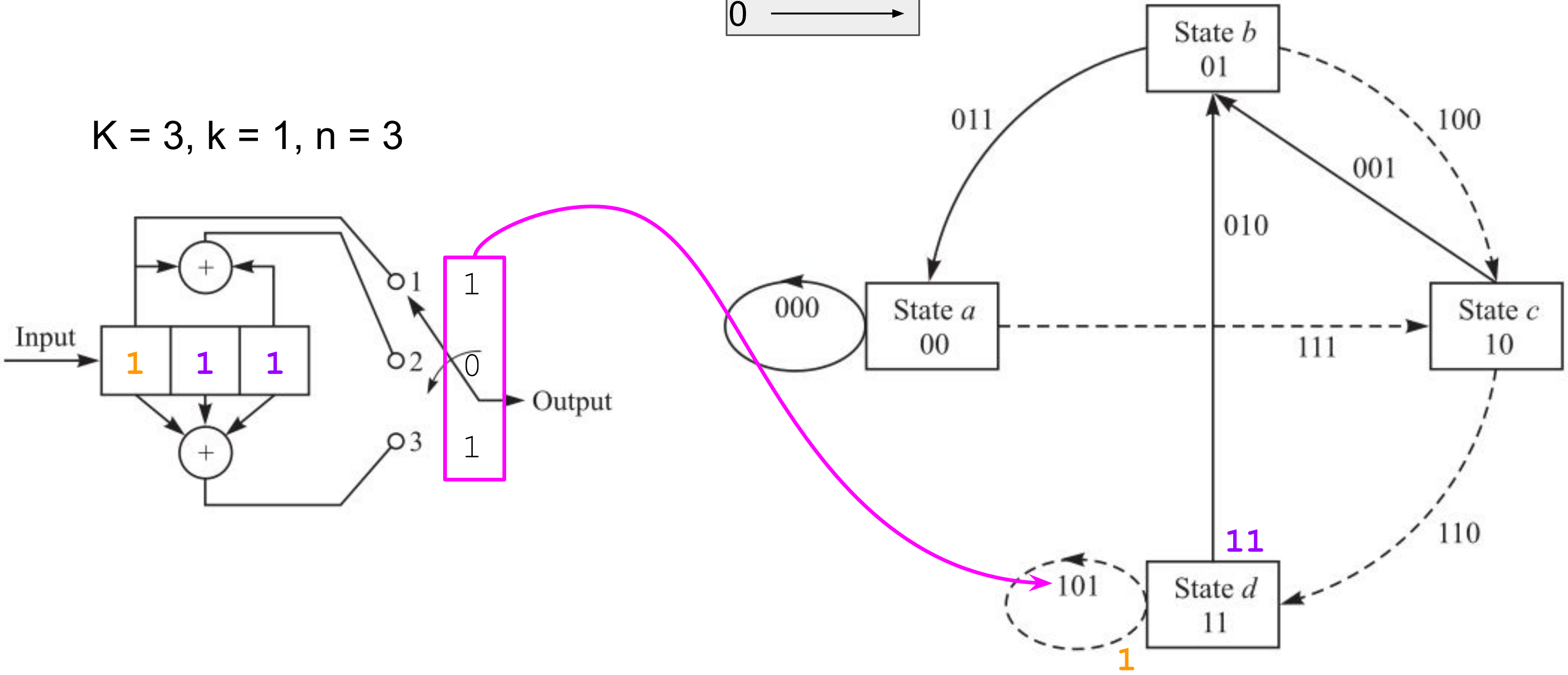


Diagrama de estados

K = 3, k = 1, n = 3



Códigos convolucionales:

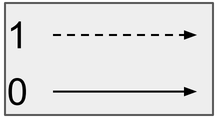
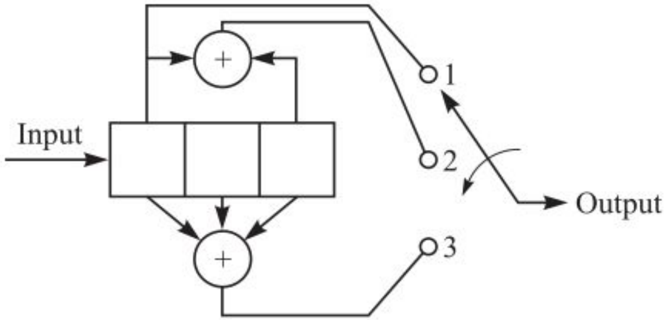
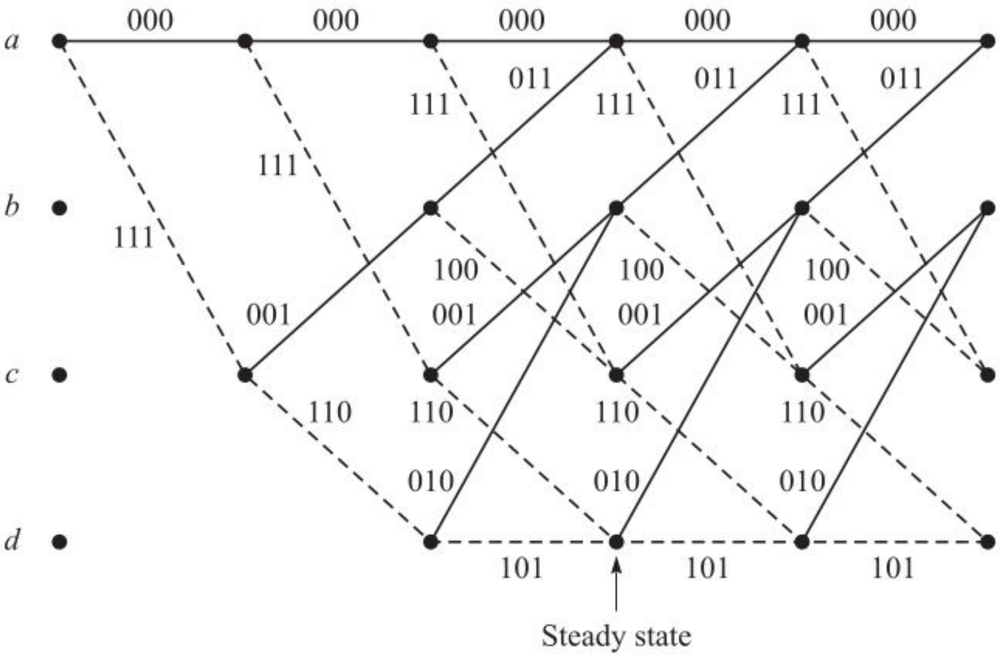


Diagrama de Trellis

K = 3, k = 1, n = 3





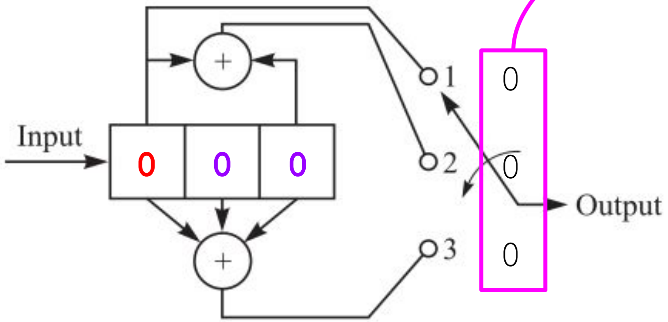
Códigos convolucionales:

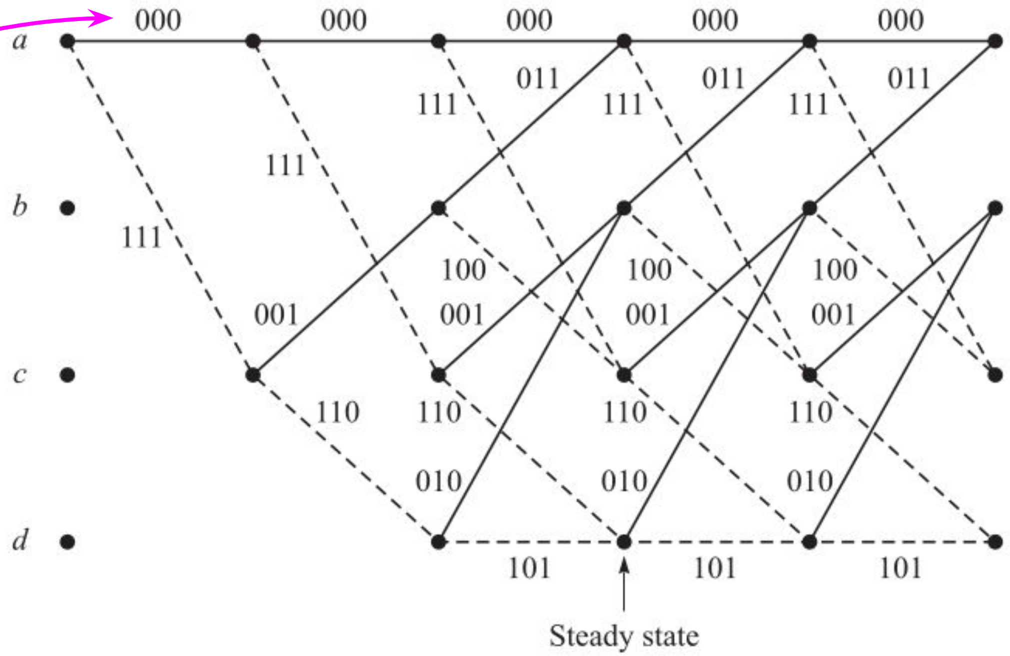
1 - - - - - - - ->
0 ———————>

<u>Diagrama de Trellis</u>

a = 00

K = 3, k = 1, n = 3





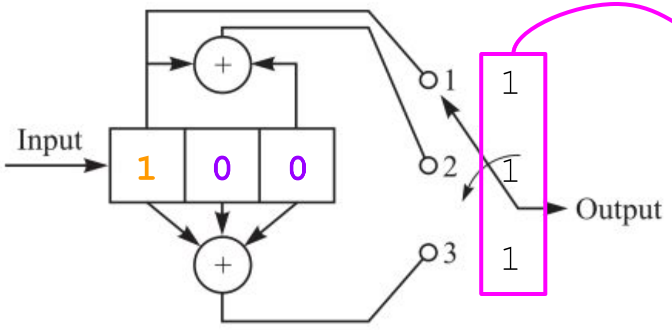
Códigos convolucionales:

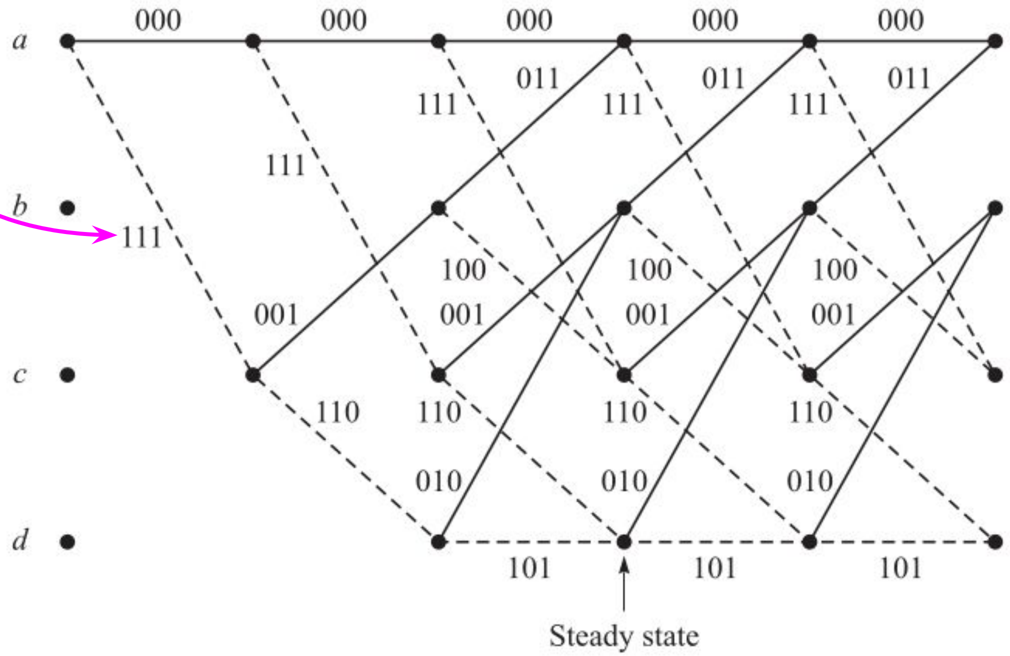
1	- - - - - - →
0	——————→

Diagrama de Trellis

a = 00

K = 3, k = 1, n = 3





Códigos convolucionales:

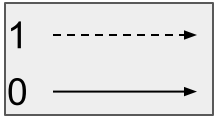
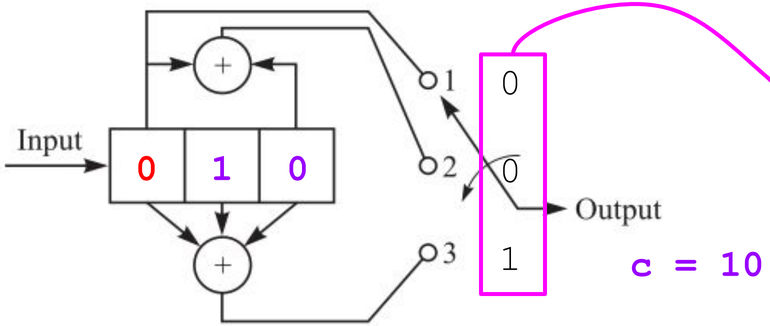
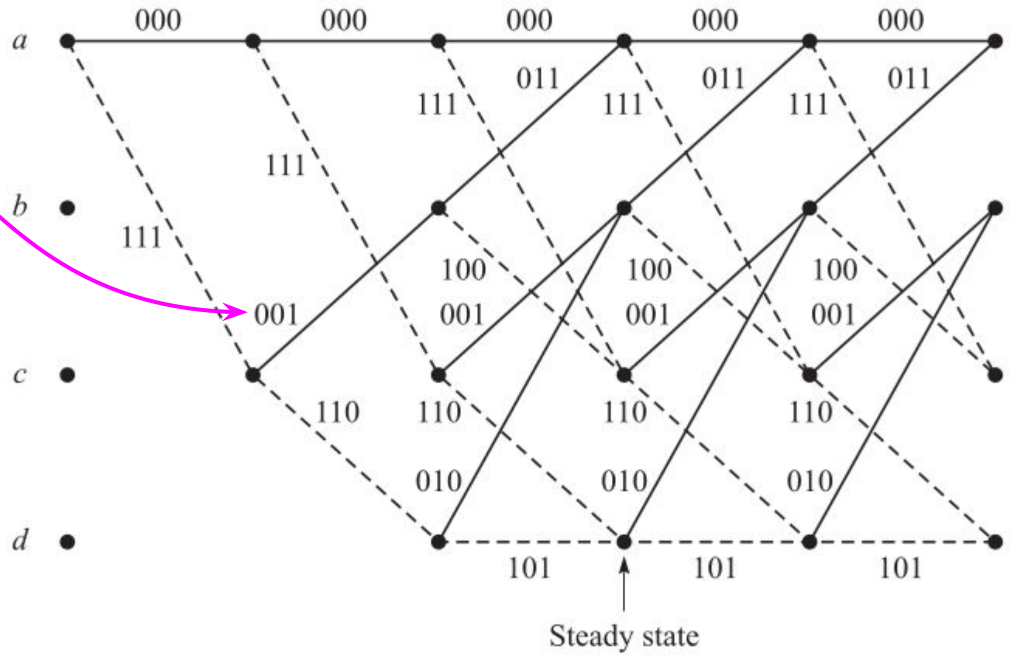


Diagrama de Trellis

K = 3, k = 1, n = 3





Códigos convolucionales:

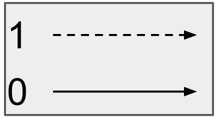
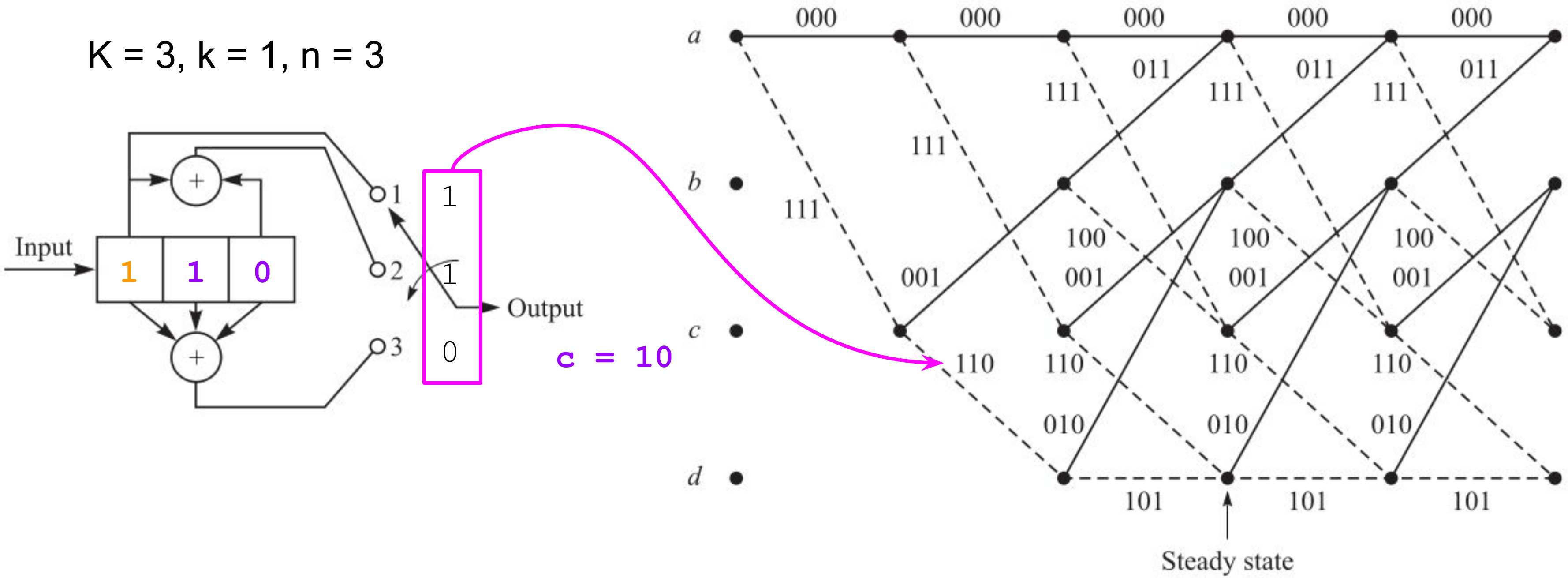


Diagrama de Trellis

K = 3, k = 1, n = 3



Algoritmo de Viterbi:

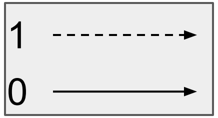
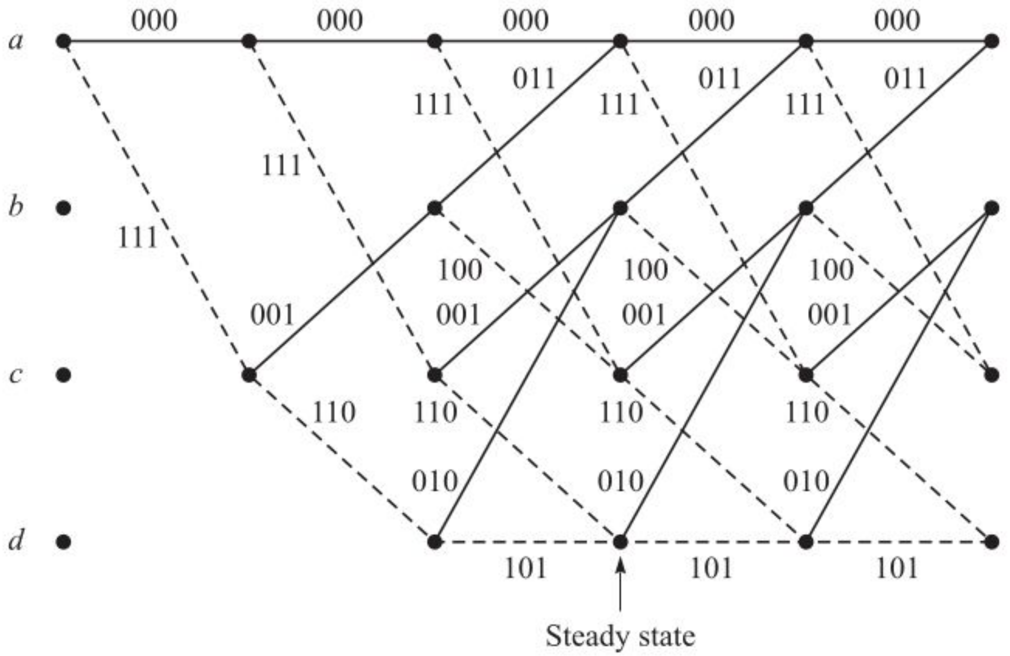


Diagrama de Trellis

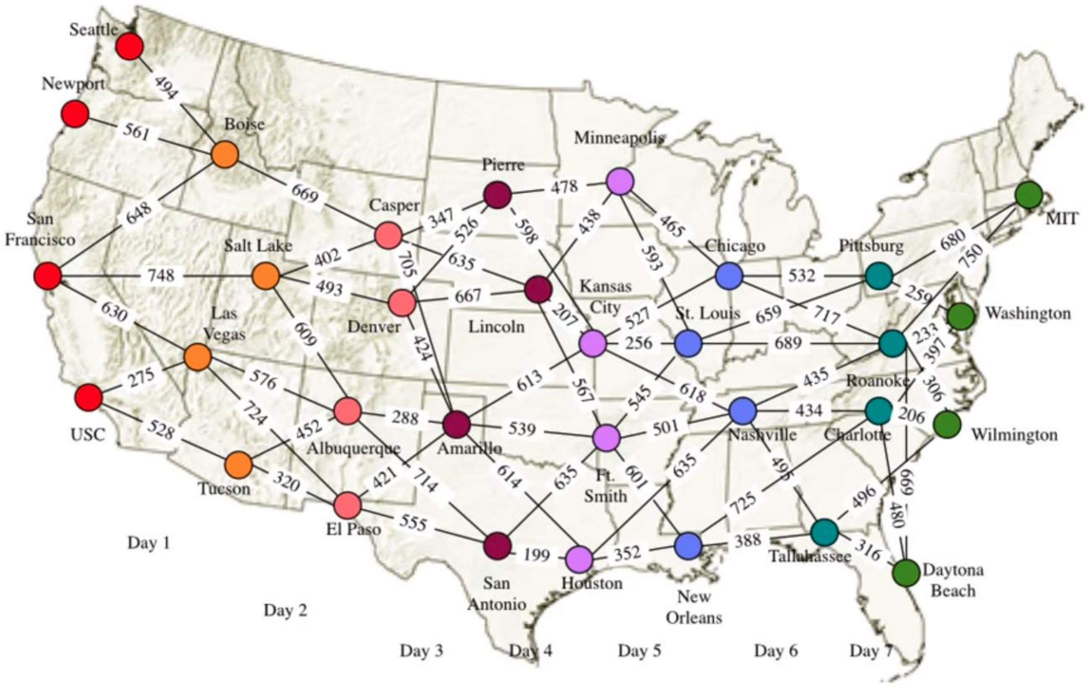
Datos	0	1	0	1	1
TX	000	111	001	100	110
Estado	00	10	01	10	11
	a	c	b	c	d
RX	010	110	011	100	100

¿Cómo se realiza la decodificación?

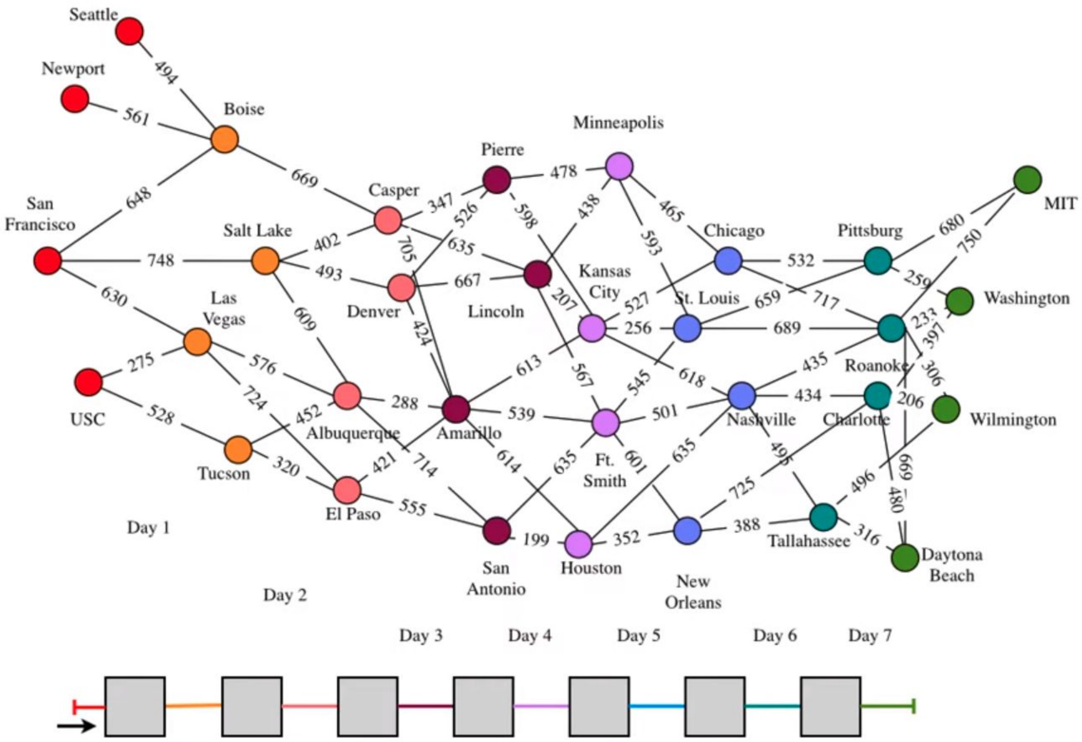
Con el algoritmo de Viterbi.



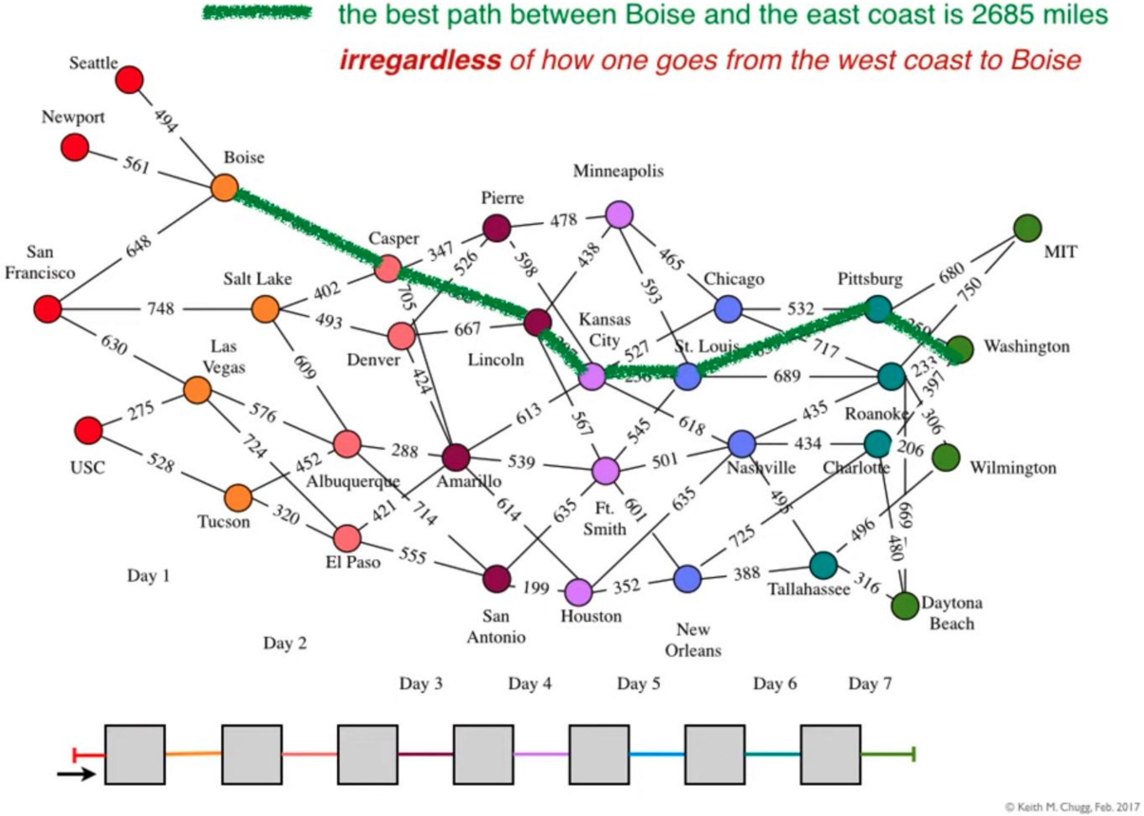
Algoritmo de Viterbi:



Algoritmo de Viterbi:

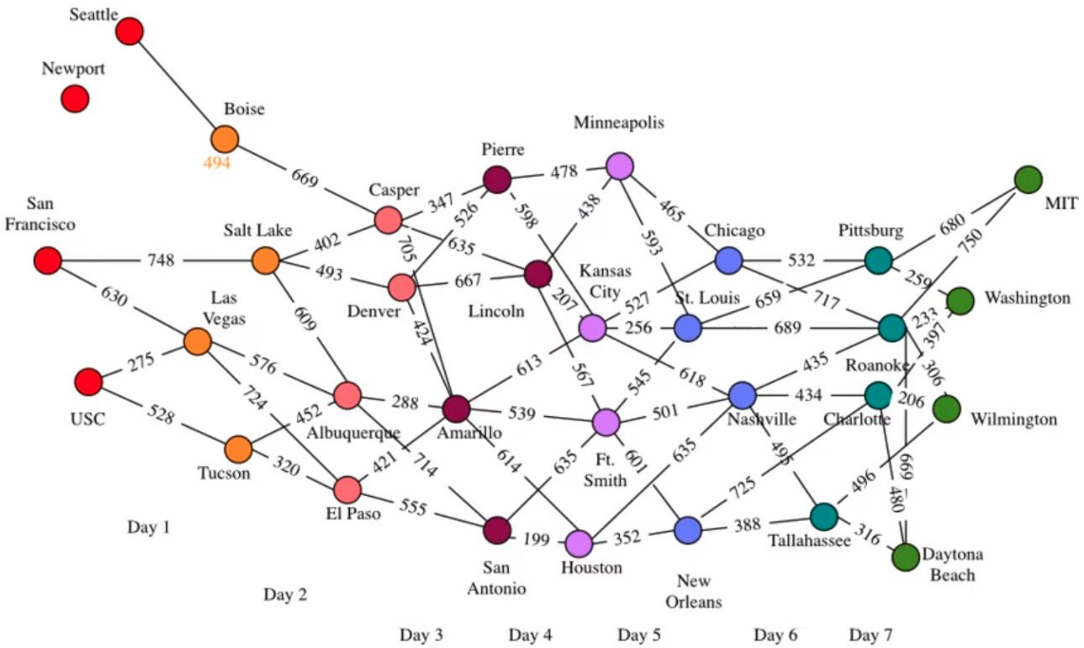


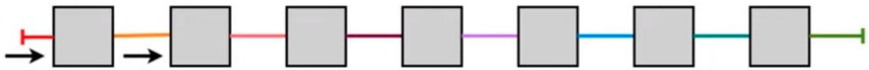
Algoritmo de Viterbi:



Algoritmo de Viterbi:

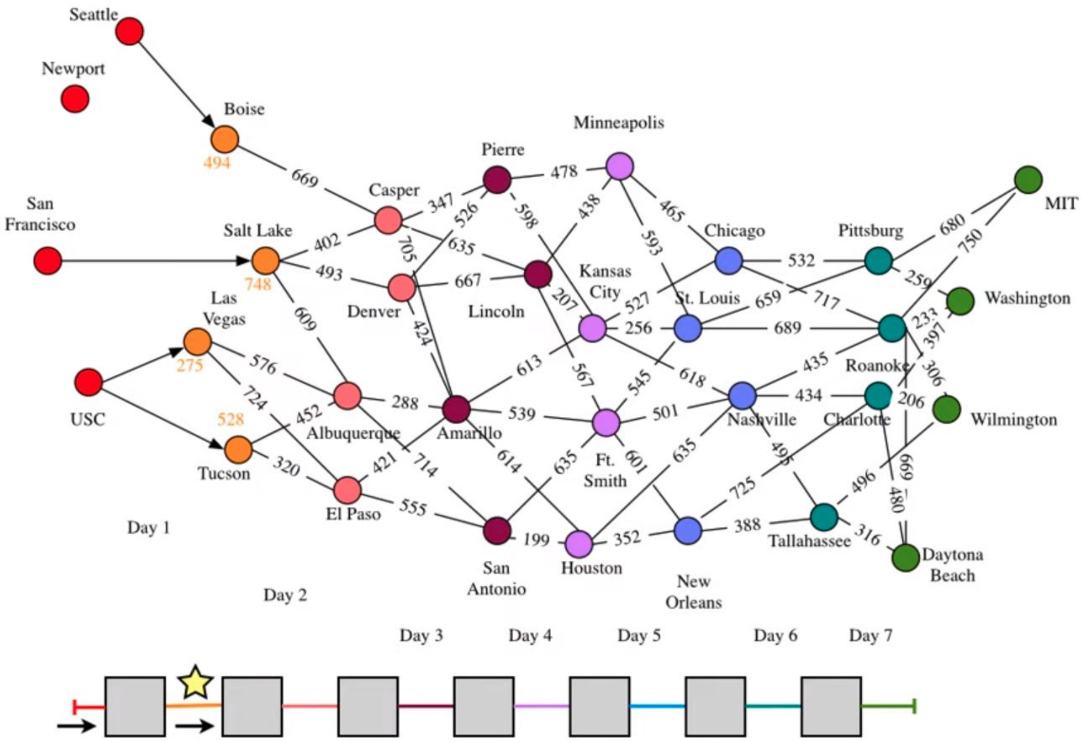
If best path from west coast to east coast, passes through Boise, it must coincide with best path from west coast to Boise





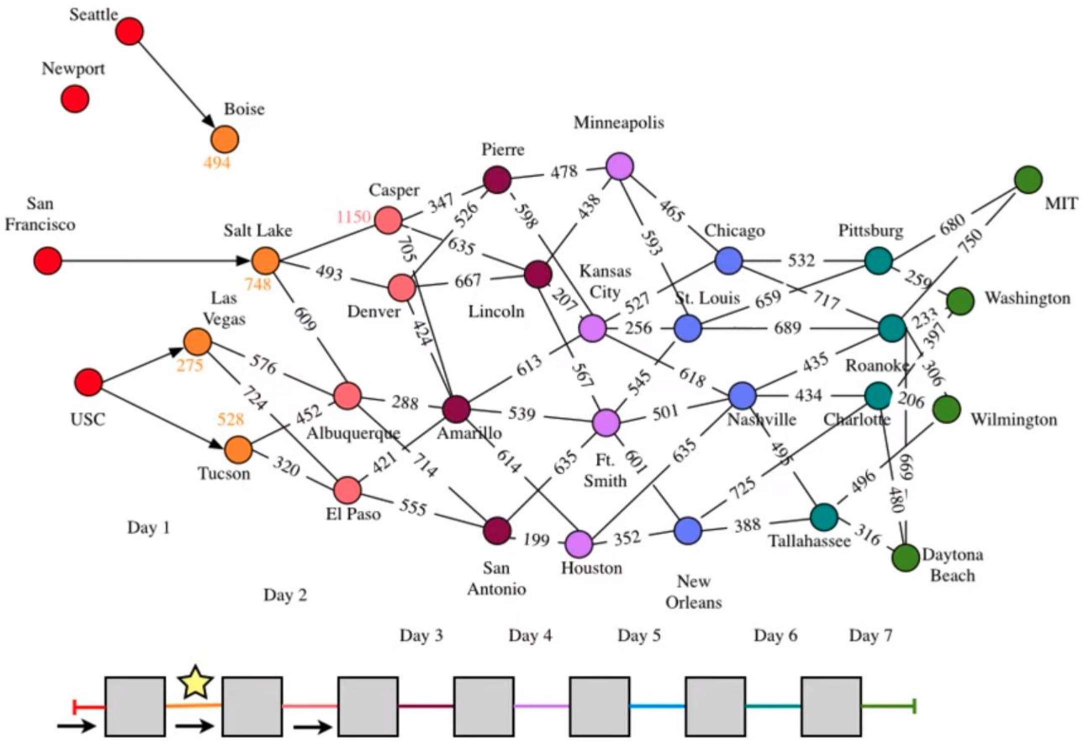
Algoritmo de Viterbi:

Repeat Boise-argument for Salt Lake, Las Vegas, Tucson

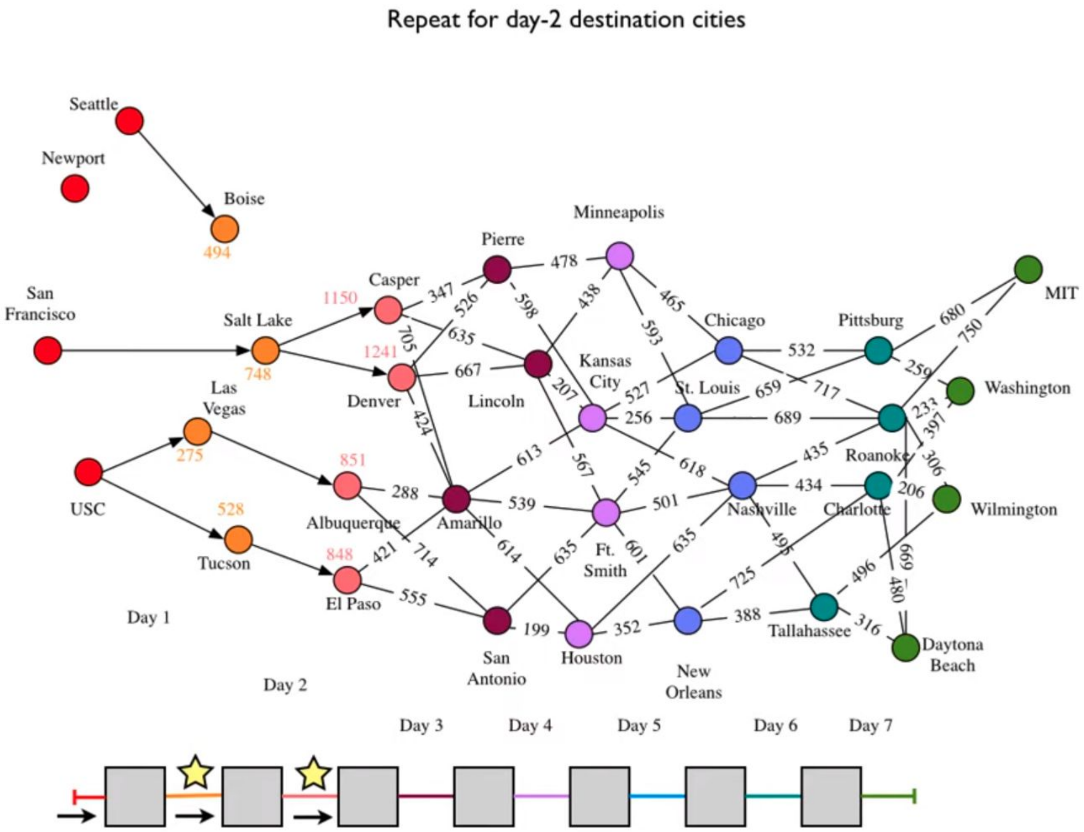


Algoritmo de Viterbi:

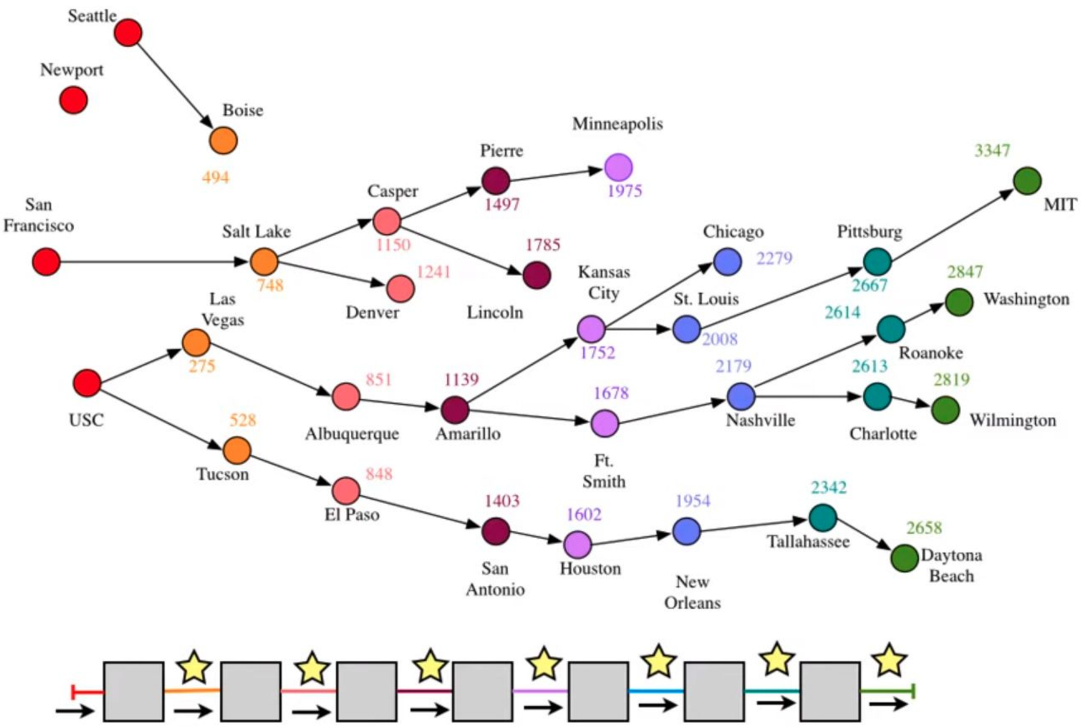
Repeat for Casper



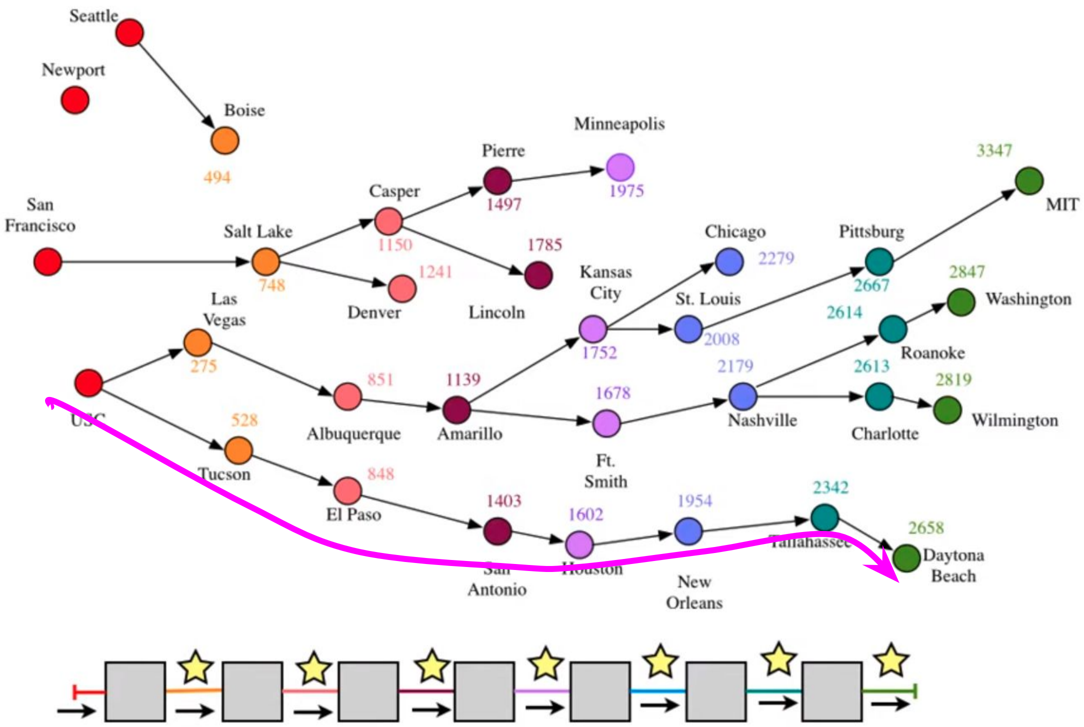
Algoritmo de Viterbi:



Algoritmo de Viterbi:



Algoritmo de Viterbi:



Algoritmo de Viterbi:

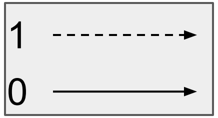
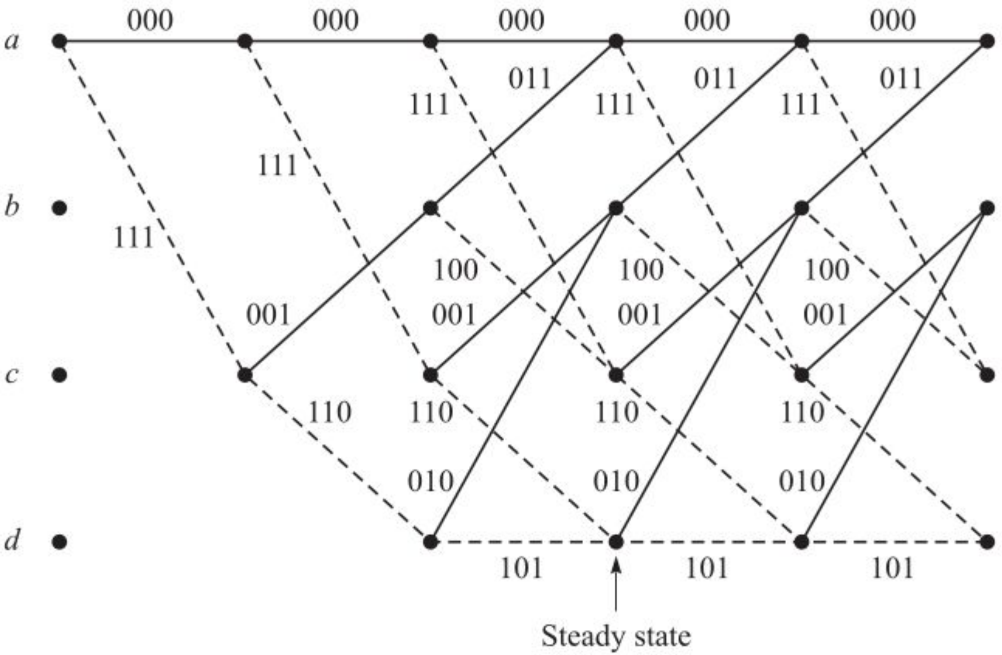


Diagrama de Trellis

```
Datos  |  0  |  1  |  0  |  1  |  1  |
TX     | 000 | 111 | 001 | 100 | 110 |
Estado | 00  | 10  | 01  | 10  | 11  |
       |  a  |  c  |  b  |  c  |  d  |
       |     |     |     |     |     |
RX     | 000 | 111 | 001 | 100 | 110 |
```



¿Cómo se realiza la decodificación?

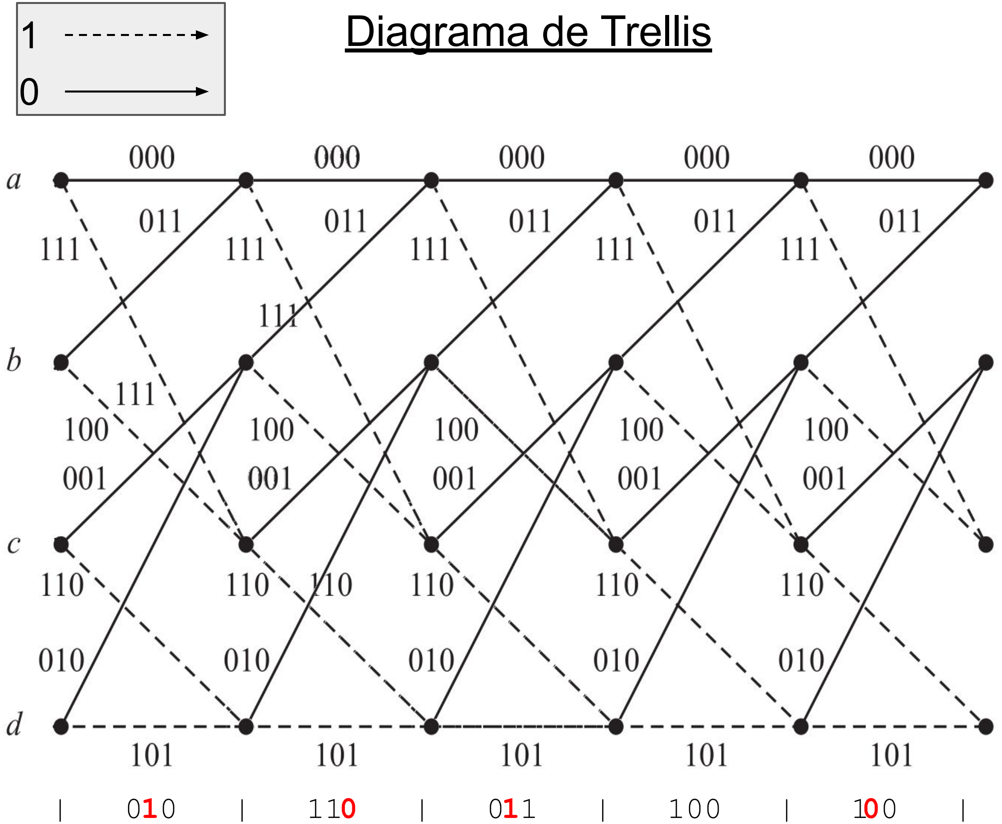
Con el algoritmo de Viterbi.

Algoritmo de Viterbi:

Distancias:

RX	010	110	011	100	100
000	1	2	2	1	1
001	2	3	1	2	2
010	0	1	1	2	2
011	1	2	0	3	3
100	2	1	3	0	0
101	3	2	2	1	1
110	1	0	2	1	1
111	2	1	1	2	2

Diagrama de Trellis



Algoritmo de Viterbi:

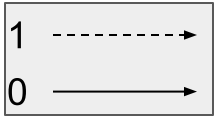
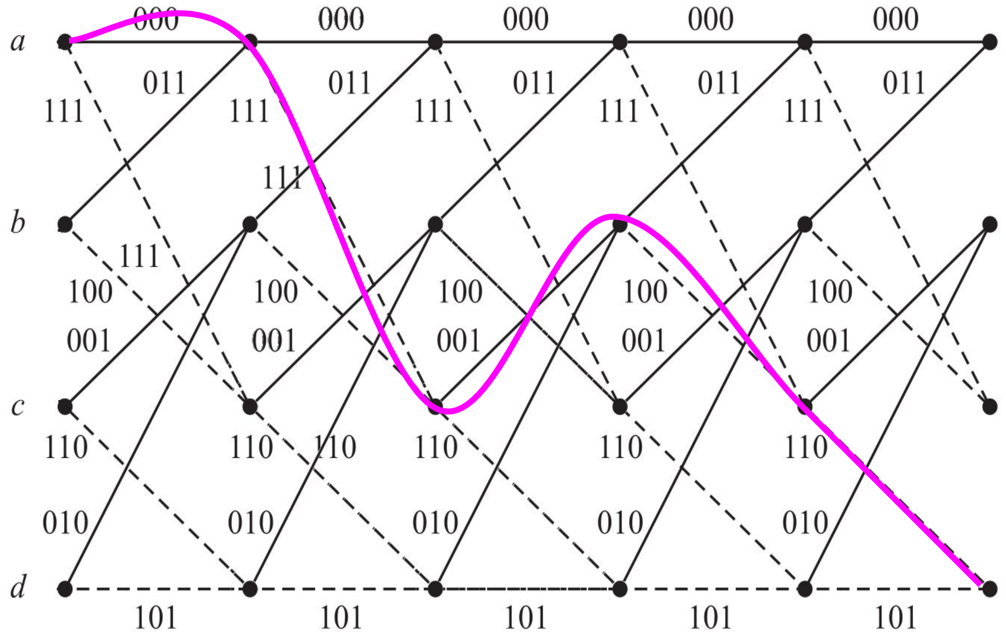


Diagrama de Trellis

<u>Distancias:</u>

RX	010	110	011	100	100
000	1	2	2	1	1
001	2	3	1	2	2
010	0	1	1	2	2
011	1	2	0	3	3
100	2	1	3	0	0
101	3	2	2	1	1
110	1	0	2	1	1
111	2	1	1	2	2

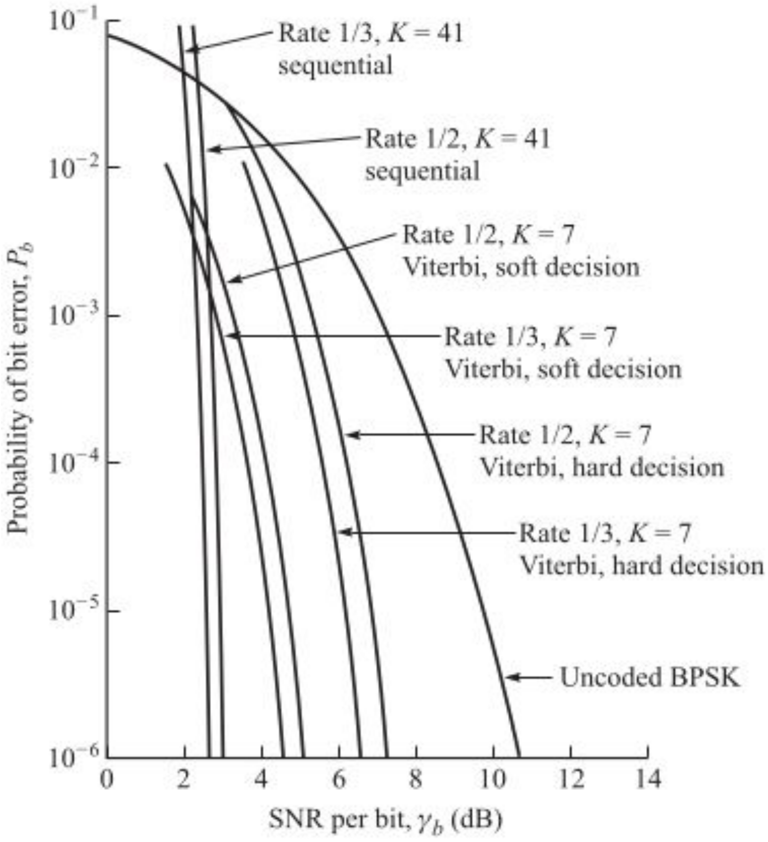


| 010 | 110 | 011 | 100 | 100 |

Códigos convolucionales:

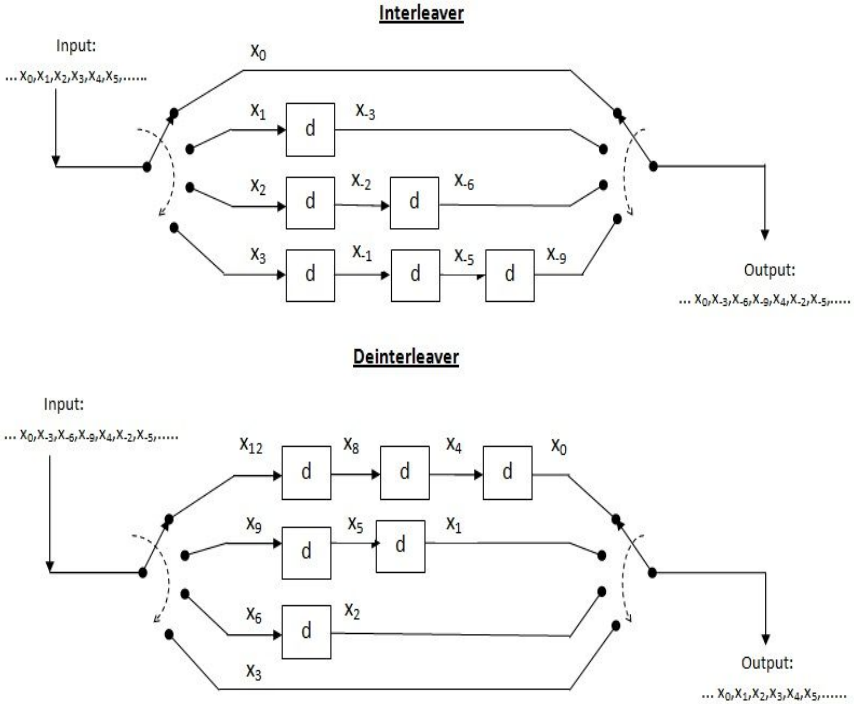
Algunos decodificadores:

- Decodificador Viterbi
    - Decisión Soft
    - Decisión Hard
- Decodificador Secuencial
- Otros decodificadores



Entrelazado:

Hace más robusta la detección y corrección de errores en presencia de ruido tipo ráfaga.



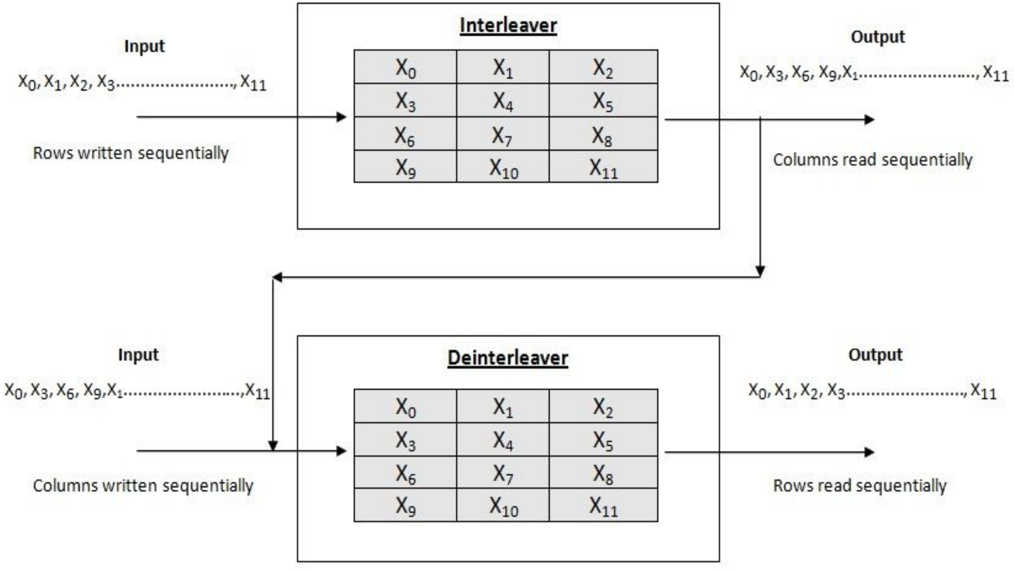
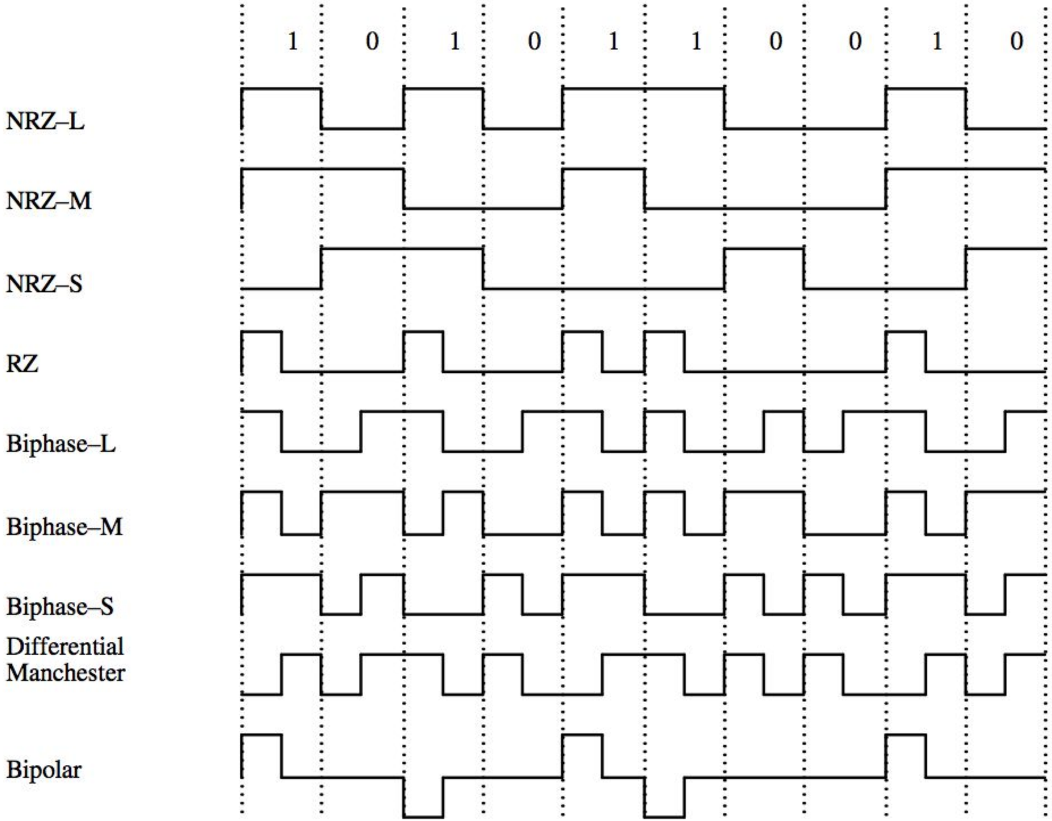


Fig. A 4 X 3 Interleaver and Deinterleaver

Codificación de línea:

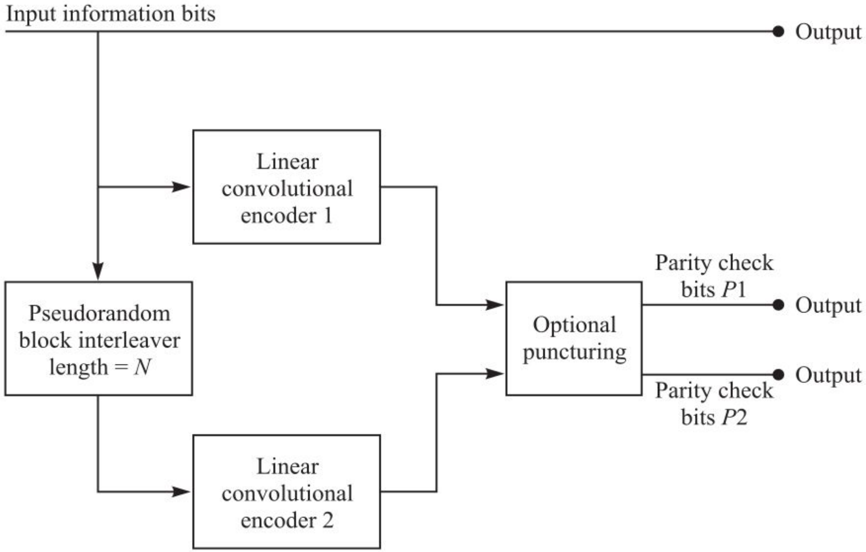
- Minimizar el Hardware
- Facilitar la sincronización
- Facilitar la detección y corrección de errores
- Modificar la densidad espectral
- Eliminar la componente de continua



Más sobre códigos:

Los distintos códigos que se presentaron se pueden combinar para obtener nuevos códigos con mejores características.

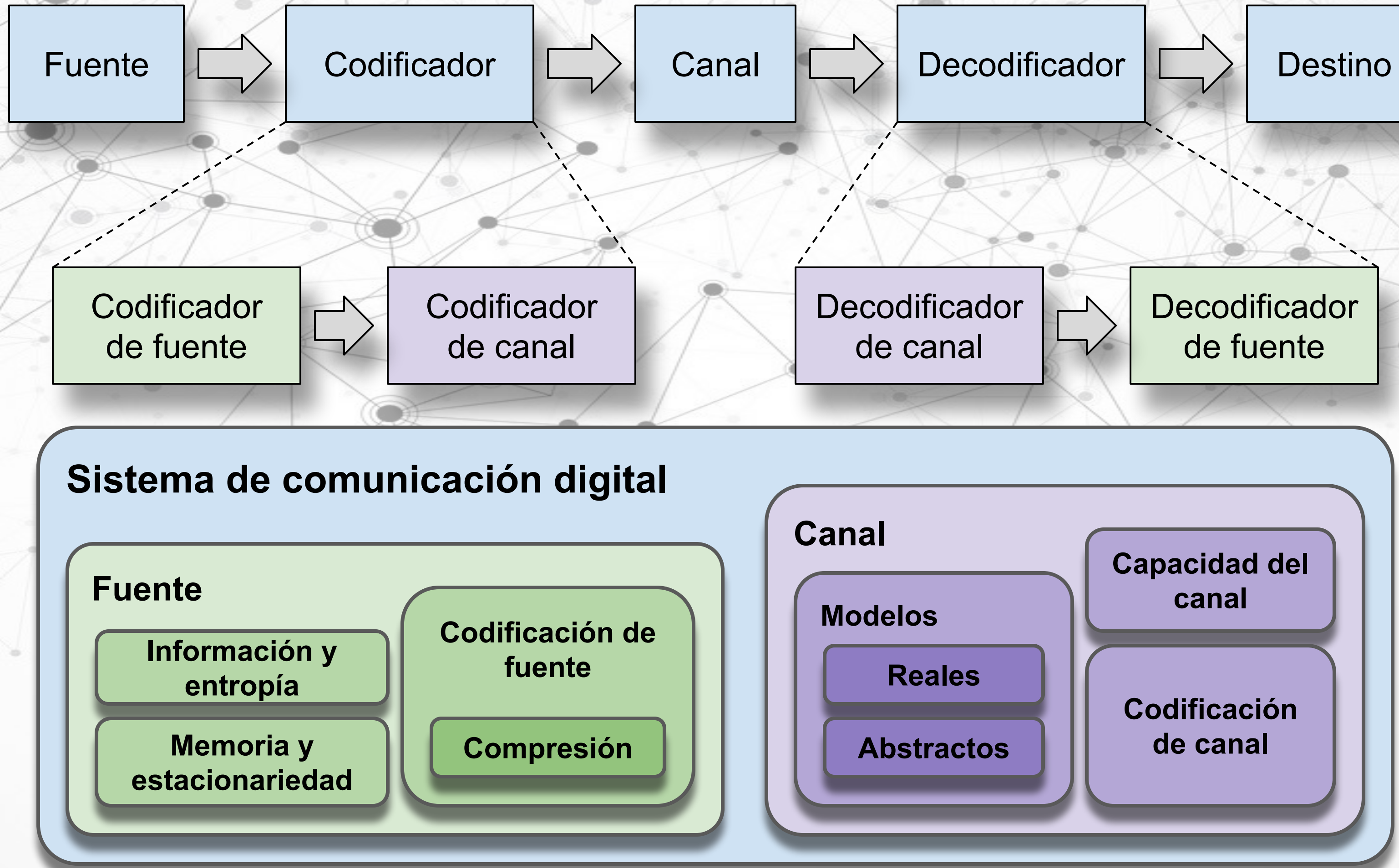
En general el gran problema para alcanzar los límites es complejidad de los decodificadores particularmente para códigos largos.



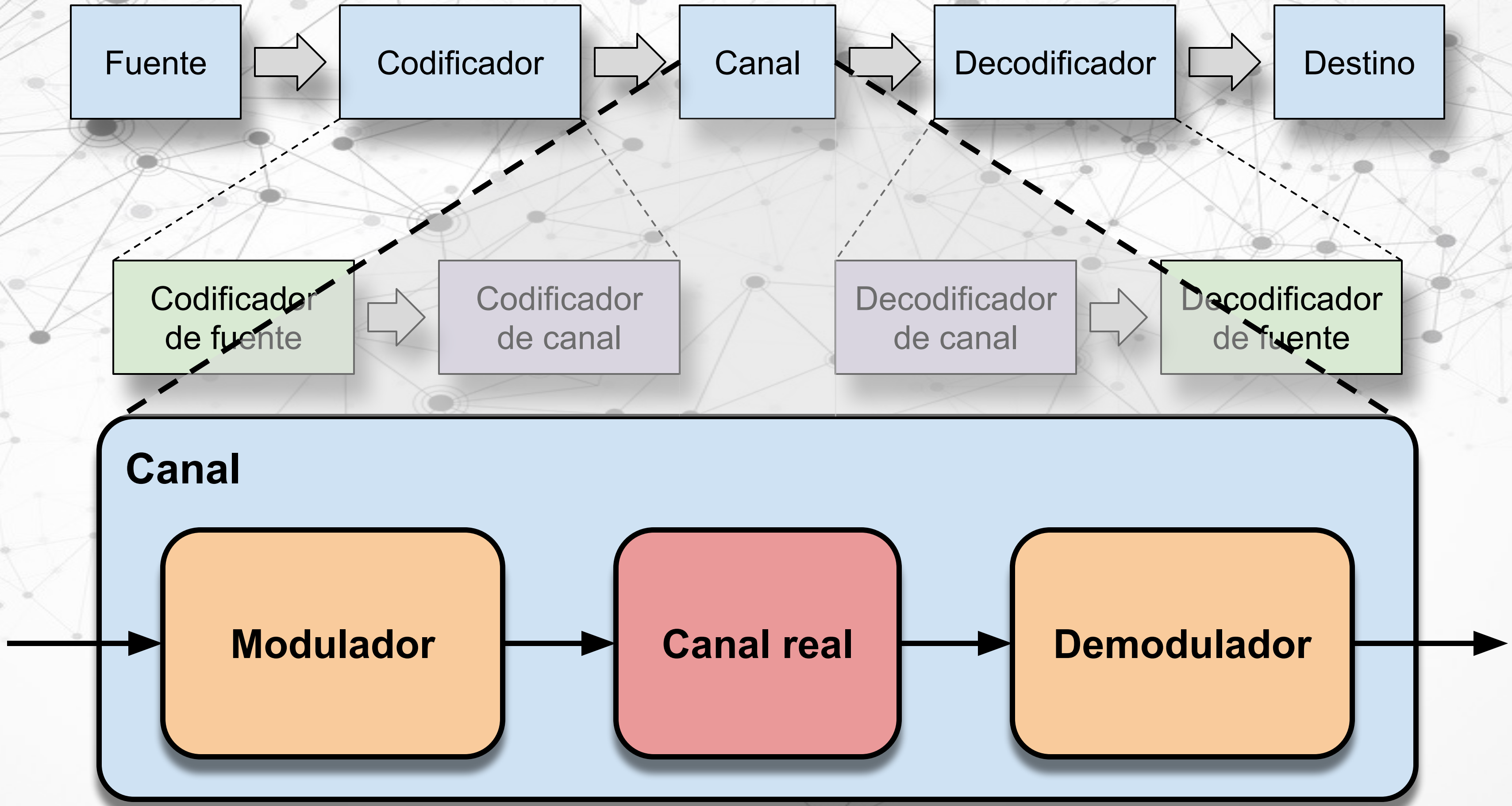
Turbo Codes (para investigar)

El mecanismo de decodificación de los códigos turbo también impulsaron a los LDPC.

Resumen y siguientes pasos



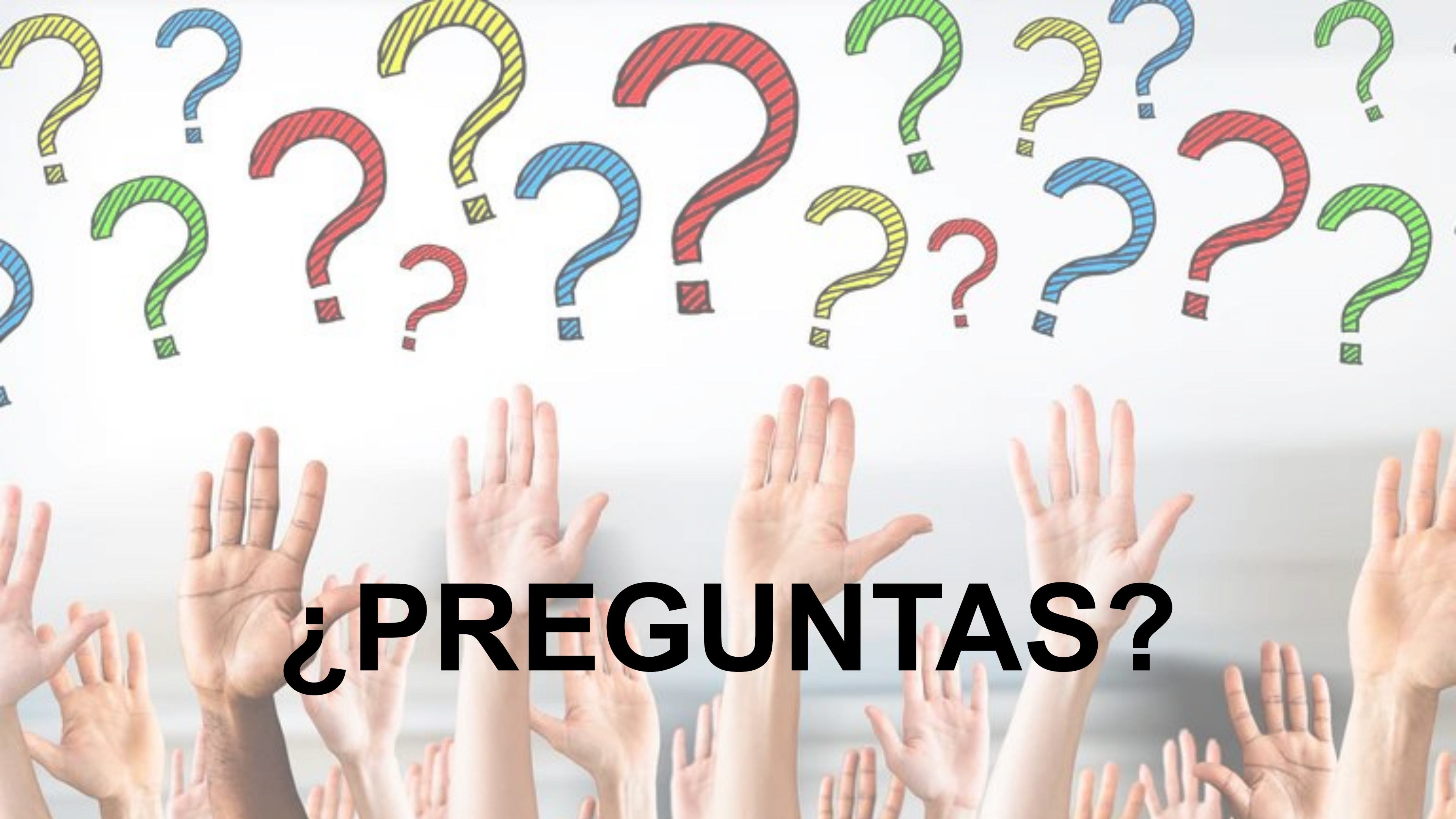
Resumen y siguientes pasos



Bibliografía y fuentes

Bibliografía:

- “*Digital Communication*”, John G. Proakis, Masoud Salehi, 5th edition.
- Andrew J. Viterbi, "*Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm*," IEEE Trans. Information Theory, vol. 13, pp. 259-260, April 1967.
- Thomas M. Cover and Joy A. Thomas. 2006. “*Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*”. Wiley-Interscience, USA.
- B. P. Lathi. 1998. “*Modern Digital and Analog Communication Systems 3e Osece*” (3rd. ed.). Oxford University Press, Inc., USA.
- Steven W. Smith. 1997. “*The scientist and engineer's guide to digital signal processing*”. California Technical Publishing, USA.
- Bernard Sklar. 1988. “*Digital communications: fundamentals and applications*”. Prentice-Hall, Inc., USA.
- Ejemplo Viterbi: https://www.youtube.com/watch?v=6JVqutwtzmo


¿PREGUNTAS?

¡Gracias!