Module:
Transmission numérique et optique

Hadj Brahim Abderrahmene

Plan de cours

- **Chapitre 1 :** Information et codage.
- **Chapitre 2 :** Transmission numérique.
- **Chapitre 3 :** Techniques de multiplexage.
- **Chapitre 4 :** Transmission optique.

Chapitre 1 : Information et codage

- Introduction.
- Codage source.
- Exemples sur le codage source.
- Codage canal.
- Exemples sur le codage canal.

Chapitre 1: Information et codages

Le rôle des télécommunications numériques est de transporter à distance un signal sous forme analogique ou numérique, cela se fait par traitement numérique de l'information.

Certaines informations sont sous forme analogique (les signaux varient continuellement dans le temps et prennent un nombre infini de valeurs. D'autres informations sont par nature numériques (des signaux varient de manière discrète dans le temps et qui prennent un ensemble fini de valeurs).

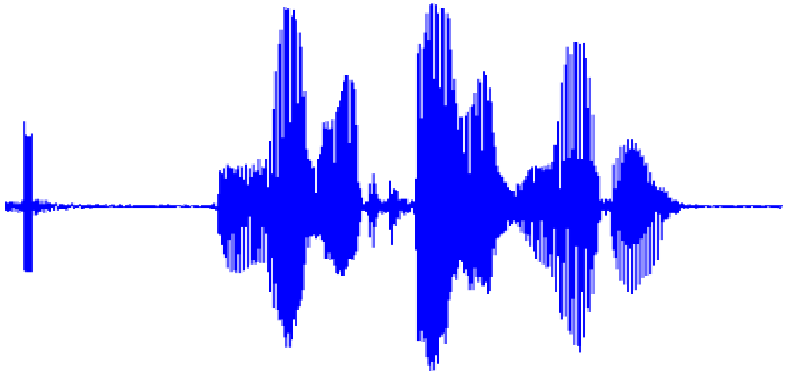
La nature des informations transmises dans télécommunications peuvent être très variée :

-Parole humaine et son haute fidélité.

-Données alphanumériques, textes et autres données structurées en un ensemble de caractères.

-Images fixes en noir et blanc ou en couleur.

-Images animées, comme des images de télévision.



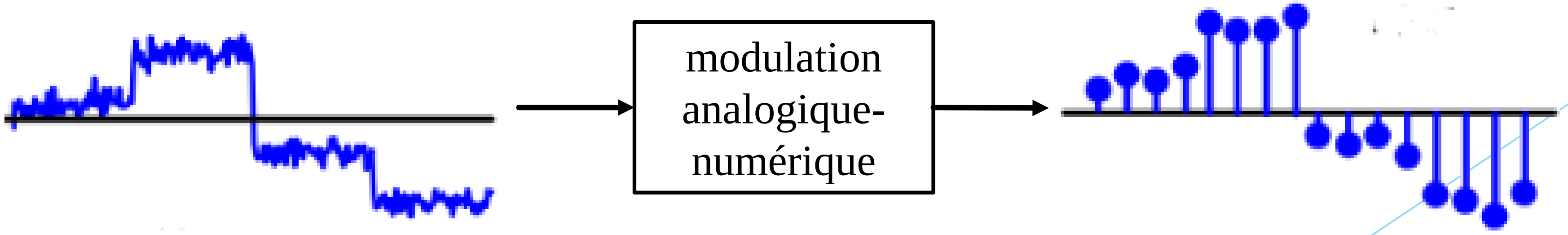
aaavcBNJMPcd

Chapitre 1: Information et codages

Un son est une onde qui se propage grâce à la vibration de la matière (solide, liquide, gaz), cette onde varie continuellement dans le temps.

Dans la transmission numérique, le signal sonore est converti en signal numérique par utiliser la modulation analogique-numérique pour obtenir un signal numérique.

La plage de l'oreille humaine est généralement comprise entre 20 Hz et 20 KHz, mais dans la transmission numérique, il est possible de la numériser à beaucoup plus faible débit (13 Kbit/s pour GSM, 8 Kbit/s pour les codeurs récents, et 2,4 Kbit/s pour des applications militaires).



Chapitre 1: Information et codages

Dans les données alphanumériques il y a des chiffres, des lettres, des signes de ponctuations et des caractères spéciaux.

Dans la transmission numérique, chaque symbole a son propre code binaire, et il change selon le codage utilisés.

Les codes alphanumériques les plus utilisés sont code ASCII (American Standard Code for Information Interchange), le code Baudot, et le code BCD (Binary Coded Decimal) ….

Decimal	Binary	ASCII	Decimal	Binary	ASCII	Decimal	Binary	ASCII
32	00100000	SP	64	01000000	@	96	01100000	`
33	00100001	!	65	01000001	A	97	01100001	a
34	00100010	"	66	01000010	B	98	01100010	b
35	00100011	#	67	01000011	C	99	01100011	c
36	00100100	$	68	01000100	D	100	01100100	d
37	00100101	%	69	01000101	E	101	01100101	e
38	00100110	&	70	01000110	F	102	01100110	f
39	00100111	'	71	01000111	G	103	01100111	g
40	00101000	(	72	01001000	H	104	01101000	h
41	00101001	)	73	01001001	I	105	01101001	i
42	00101010	*	74	01001010	J	106	01101010	j
43	00101011	+	75	01001011	K	107	01101011	k
44	00101100	,	76	01001100	L	108	01101100	l
45	00101101	-	77	01001101	M	109	01101101	m
46	00101110	.	78	01001110	N	110	01101110	n
47	00101111	/	79	01001111	O	111	01101111	o
48	00110000	0	80	01010000	P	112	01110000	p
49	00110001	1	81	01010001	Q	113	01110001	q
50	00110010	2	82	01010010	R	114	01110010	r
51	00110011	3	83	01010011	S	115	01110011	s
52	00110100	4	84	01010100	T	116	01110100	t
53	00110101	5	85	01010101	U	117	01110101	u
54	00110110	6	86	01010110	V	118	01110110	v
55	00110111	7	87	01010111	W	119	01110111	w
56	00111000	8	88	01011000	X	120	01111000	x
57	00111001	9	89	01011001	Y	121	01111001	y
58	00111010	:	90	01011010	Z	122	01111010	z
59	00111011	;	91	01011011	[	123	01111011	{
60	00111100	<	92	01011100	\	124	01111100	|
61	00111101	=	93	01011101	]	125	01111101	}
62	00111110	>	94	01011110	^	126	01111110	~
63	00111111	?	95	01011111	_	127	01111111	DEL

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

CARACTÈRES	CODE	CARACTÈRE	CODE
0	00 0000	Blanc	01 0000
1	00 0001	/	01 0001
2	00 0010	S	01 0010
3	00 0011	T	01 0011
4	00 0100	U	01 0100
5	00 0101	V	01 0101
6	00 0110	W	01 0110
7	00 0111	X	01 0111
8	00 1000	Y	01 1000
9	00 1001	Z	01 1001
Espace	00 1010	≠	01 1010
=	00 1011		01 1011
Apostrophe	00 1100	(	01 1100
>	00 1111	-	01 1111
√	00 1110	/	01 1110
		Annulation	01 1111
_	10 0000	+	11 0000
J	10 0001	A	11 0001
K	10 0010	B	11 0010
L	10 0011	C	11 0011
M	10 0100	D	11 0100
N	10 0101	E	11 0101
O	10 0110	F	11 0110
P	10 0111	G	11 0111
Q	10 1000	H	11 1000
R	10 1001	I	11 1001
!	10 1010	?	11 1010
S	10 1011	.	11 1011
*	10 1100	)	11 1100
]	10 1111	[	11 1111
;	10 1110	<	11 1110
Δ	10 1111		11 1111

Chapitre 1: Information et codages

Un appareil photographique numérique assure la numérisation d'une image, cela se fait par découper l'image en trame.

L'image numérique est une image décrite en termes de lignes et chaque ligne plusieurs points (pixels), qui attribuent un nombre binaire correspondant à la couleur de la case.

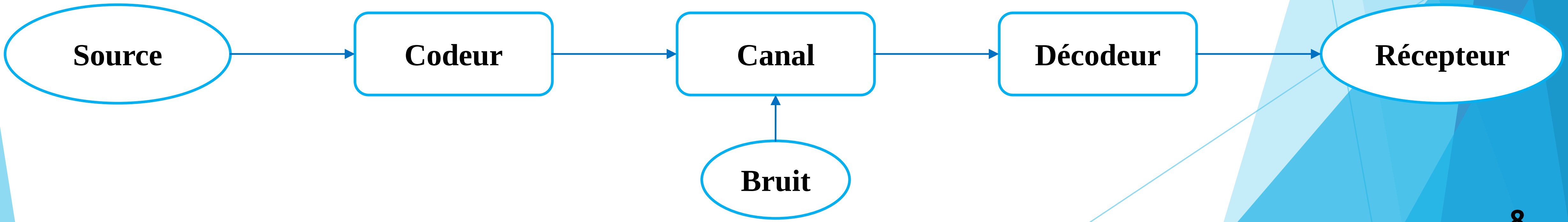
Par exemple: une image de 240 x 640 pixels occupe un volume 19200 octets pour une image binaire, occupe un volume 153600 octets pour une image de niveaux de gris, et , occupe un volume 460800 octets pour une image couleur.

Chapitre 1: Information et codages

La théorie des transmission numérique s'intéresse aux moyens de transmettre une information depuis la source jusqu’à un utilisateur à travers un canal.

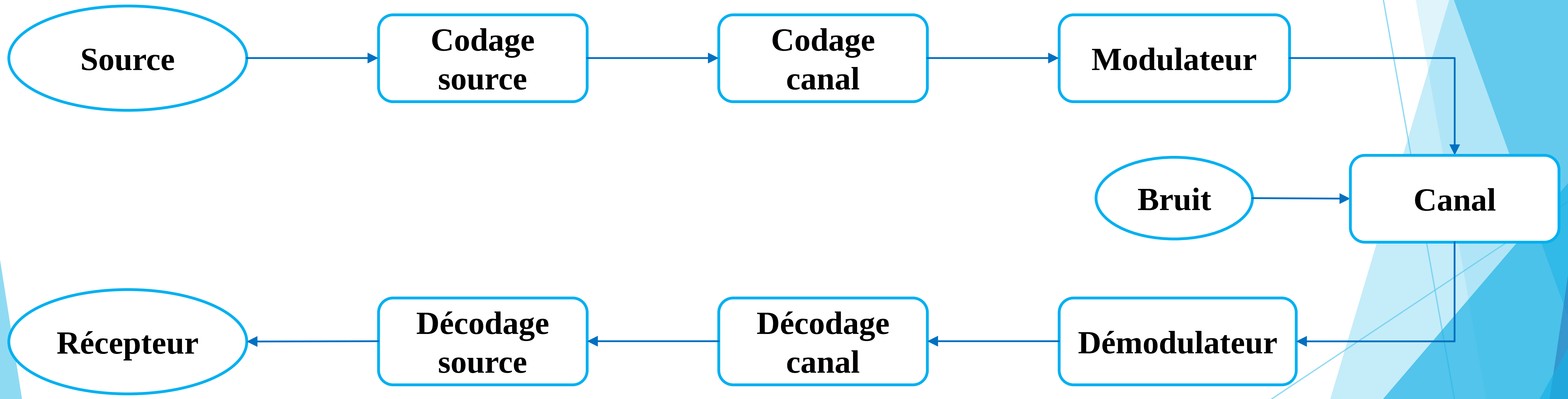
Avant la transmission numérique, le codeur effectuée l'ensemble des opérations sur la sortie de la source pour obtenir le signal à transmettre qui est compatible avec le canal.

La théorie de l’information a été créée par C. E. Shannon dans les années 40, elle consiste en l’élaboration et l'étude de modèles pour la source et le canal qui utilisent différents outils comme les probabilités.



Chaine de transmission numérique

La chaine de transmission numérique est composée de plusieurs éléments essentiels en partant de la source de message jusqu'au récepteur.. Cette chaine est utilisé dans les téléphones portables, des ordinateurs …, pour communiquer une certaine information.



Source

En transmission numérique, il est nécessaire que les messages à transmettre soient sous forme numérique, c'est-à-dire constitués par une suite d'élément binaire. Cette suite est caractérisée par des probabilités p_0 et p_1, son débit binaire ***D*** qui défini comme le nombre d'éléments binaires qu'elle émet par unité de temps, et sa rapidité de modulation ***r***.

$$r = \frac{1}{T_b} \quad (bit/s)$$

La rapidité de modulation ***S*** est définie comme le nombre de signaux émis par le modulateur par unité de temps.

$$S = \frac{1}{T} \quad (bauds)$$

$$D = r \times \log_2(V)$$

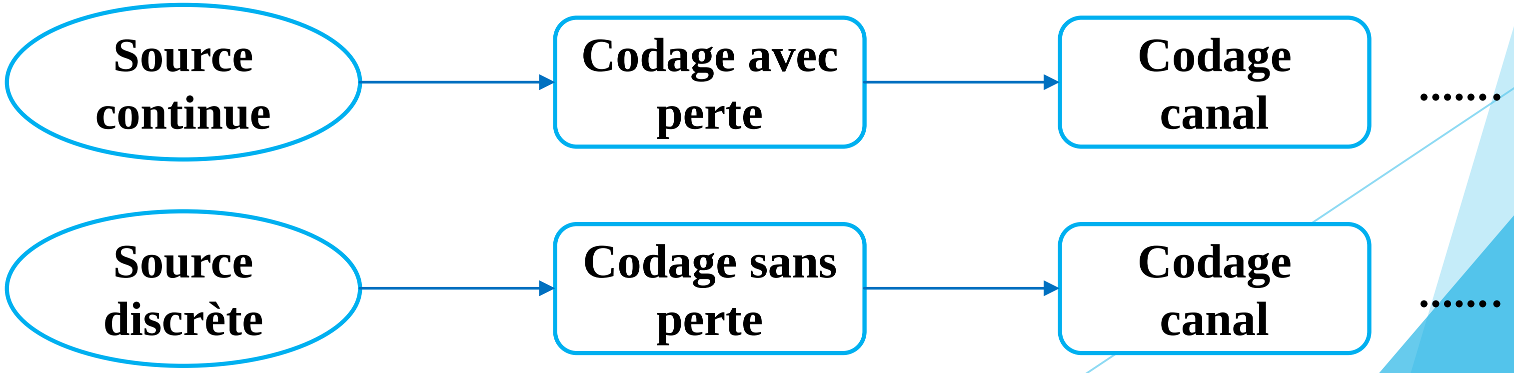
Codage source

Le codage source (ou compression des sonnées) consiste à supprimer les éléments binaires peu significatifs, qui permet de minimiser le débit binaire à transmettre.

Selon la source (l'information à transmettre); il existe deux classes de codage source : codage source avec perte, et codage source sans perte.

Le codage source avec perte est utilisé dans les sources continues (son, image,…) car il est nécessaire de quantifier les données. En plus, minimiser la distorsion entre les données originales de la source et les données reconstruites pour le récepteur.

Le codage source sans perte est utilisé dans les sources discrètes (textes…), qui repose sur la notion d'entropie de la source.



Codage source

Le codage source composé de différents symboles a plusieurs propriétés telles que :

- Chaque symbole a son propre code : $X = [x_1, x_2, , , , x_N] \Longrightarrow C = [c_1, c_2, , , , c_N]$

Exemple : message

Symbole	Code 1	Code 2	Code 3
m	01	0	000
e	000	101	1
s	110	001	01
a	0111	1111	0010
g	0100	1101	1100

- Chaque symbole a un code unique (et l'inverse)

$$X = [x_1, x_2, , , , x_N] \Longrightarrow C = [c_1, c_2, , , , c_N]$$

$$c_i \neq c_j \quad i \neq j$$

Codage source

Le codage source sans perte est basé sur les probabilités de chaque symbole de la source :

$$P_i = \frac{R(x_i)}{\sum_{j=1}^{N} R(x_j)}$$

$$X = [x_1, x_2, , , , x_N] \Longrightarrow P = [p_1, p_2, , , , p_N] \qquad \sum_{i=1}^{N} P_i = 1$$

Exemple: bbaccdabeb

Symbole	a	b	c	D	E
Répétition	2	4	2	1	1
Probabilité	0,2	0,4	0,2	0,1	0,1

Codage source

Pour déterminer si le codage source sans perte a de bonnes performances, plusieurs métriques sont prises en compte :

- **Taux de compression :** est une mesure de la réduction des données.

$$CR = \frac{N_c}{N_i} \times 100$$

- **L'entropie :** qui mesure la quantité d'information moyenne associé à l'apparition de chaque symbole

$$X = [x_1, x_2, , , , x_N] \Longrightarrow P = [p_1, p_2, , , , p_N] \Longrightarrow \quad H(X) = -\sum_{i=1}^{N} p_i \log_2 p_i$$

- **Longueur moyenne :** $L(X) = \sum_{i=1}^{N} p_i \, l_i$ $\qquad$ l_i est la longueur de code du symbole x_i

- **Efficacité du codage :** $E = \frac{H(x)}{L(x)}$

- **Redondance du codage :** $R = 1 - E$

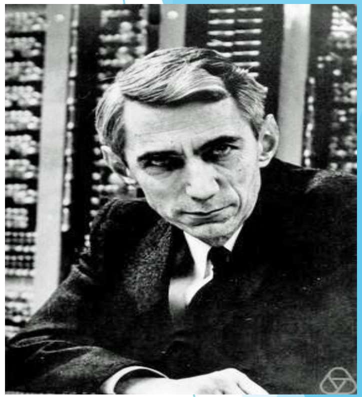
Codage source

Les algorithmes de codage source sans perte les plus utilisés sont :

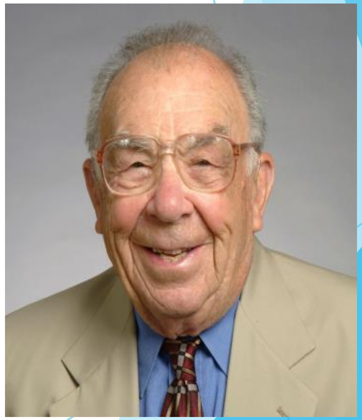
- Codage de Shannon-Fano
- Codage de Huffman
- Codage arithmétique
- Codage LZW

Codage de Shannon-Fano

- Code développé en 1960 par Claude E. Shannon et Robert M. Fano.
- Algorithme simple avec des performances élevées.
- Assignation du code selon la probabilité de chaque symbole.
- Construction d'un code préfixe basé sur le théorie de Shannon.
- Exemple d'utilisation : format ZIP.



Claude Elwood Shannon



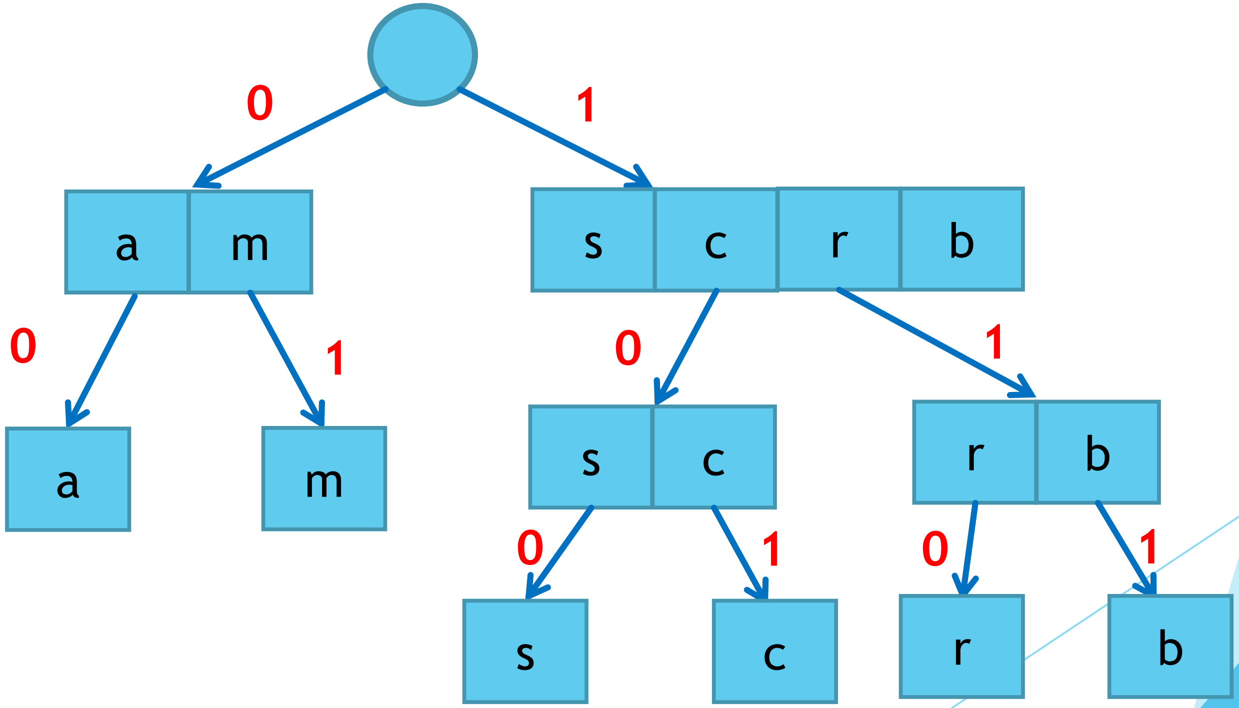
Robert Mario Fano

Codage de Shannon-Fano

1) Déterminer les probabilités de chacun des symboles.
2) Ordonner les symboles selon leurs probabilité d'apparence décroissant.
3) Calculer les différences minimales entre les probabilités.
4) Déterminer la plus petite valeur entre les différents minimales.
5) Diviser l'ensemble des symboles en deux sous-groupes selon la position de la plus petite valeur des différences minimales.
6) Assigner un '0' pour le premier sous-groupe et un '1' pour le deuxième sous-groupe.
7) Réitérer à partir la $3^{ème}$ étape en subdivisant les sous-groupes.
8) Condition d'arrêt : tous les sous-groupes ont un seul symbole.

Codage de Shannon-Fano

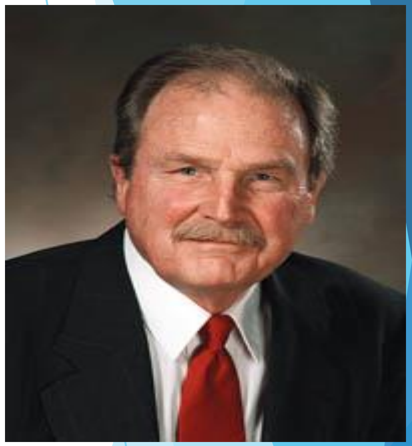
Symboles	a	b	c	m	r	s
Fréquence	9	3	6	8	5	7
Code	00	111	101	01	110	100



$$\boldsymbol{CR = 31,91\%}$$
$$\boldsymbol{H = 2,5096}$$
$$\boldsymbol{L = 2,5526}$$
$$\boldsymbol{E = 98,31\%}$$
$$\boldsymbol{R = 1,68\%}$$

Codage de Huffman

- Code développé en par David A. Huffman (étudiant de Shannon et Fano).
- Le code de Huffman est code aussi simple que le code de Shannon-Fano.
- Le code de Huffman est optimal et il est basé sur deux observations :
  - Il assigne moins de bits aux symboles les plus fréquents et plus de bits au symboles les moins fréquents.
  - Les deux moins fréquents symboles ont la même longueur de code.
- Exemple d'utilisation : format JPEG.



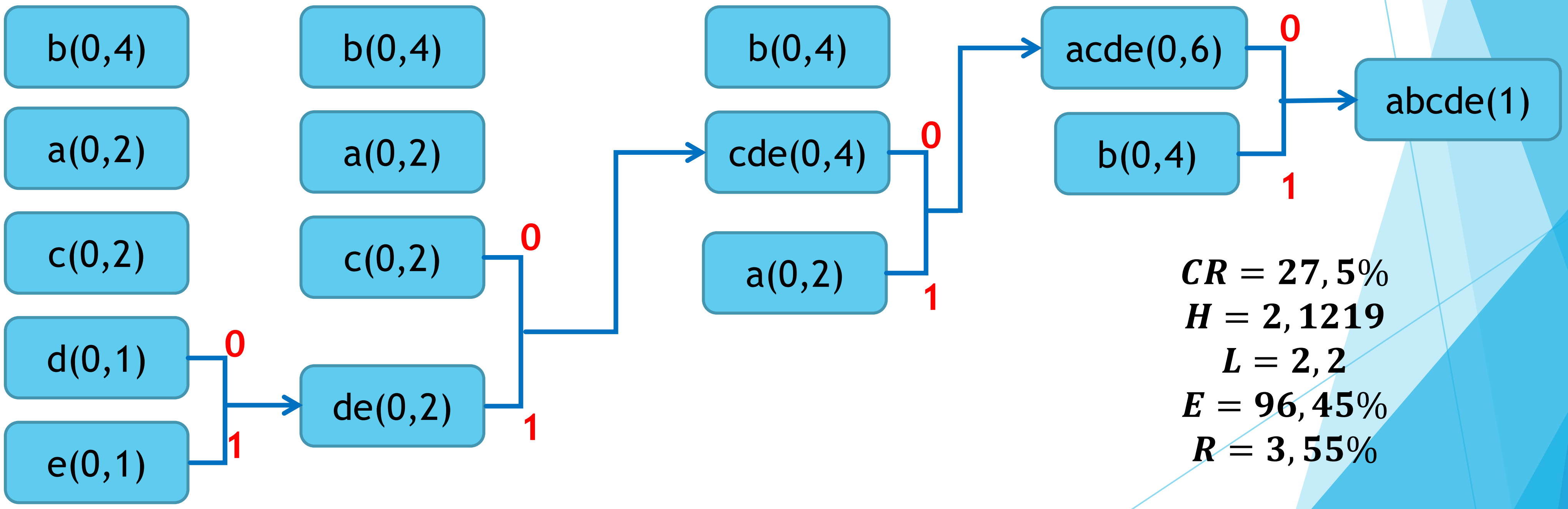
David Albert Huffman

Codage de Huffman

1) Déterminer des probabilités de chacun des symboles.
2) Ordonner les symboles selon leurs probabilité d'apparence décroissant.
3) Assigner un code '1' pour le dernier symbole et un code '0' pour l'avant-dernier symbole.
4) Combiner les deux derniers symboles en un seul sous-groupe.
5) Réitérer à partir la $2^{ème}$ étape sur le nouveau groupe.
6) Condition d'arrêt : tous les symboles sont combinés en un seul sous-groupe.

Codage de Huffman

Symboles	a	b	c	d	e
Fréquence	2	4	2	1	1
Code	01	1	000	0010	0011



$$CR = 27,5\%$$
$$H = 2,1219$$
$$L = 2,2$$
$$E = 96,45\%$$
$$R = 3,55\%$$

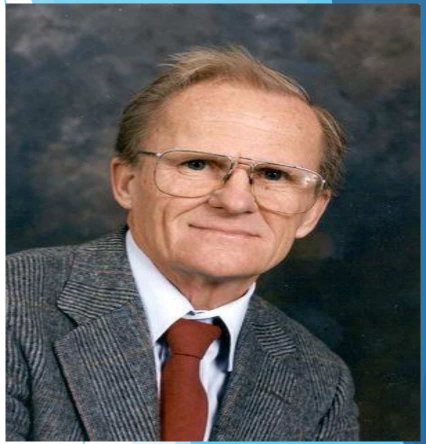
Codage adaptatif de Huffman

- Codage de Huffman nécessite une connaissance à priori de la probabilité d'apparition des symboles.
- Le codage adaptatif de Huffman, ou codage dynamique de Huffman, est basé sur la mise à jour du code du symbole à chaque itération.
- Chaque symbole a son code initial qui peut être modifié lors de l'utilisation du codage.
- Le codage adaptatif de Huffman est utilisé pour le message qui a différentes fréquences de symboles dans différentes régions.

1) Calculer le nombre totale des nœuds possibles dans l'arbre par $(2 \times n) - 1$.
2) Initialiser chaque symbole par un code initiale.
3) Changer le code des symboles en fonction du message à transmettre.

Codage arithmétique

- Le codage arithmétique pratique a vu le jour en 1976 grâce aux travaux de Jorma J. Rissanen et Richard C. Pasco.
- Le codage arithmétique est un codage de compression populaire après le codage de Huffman et il est particulièrement utile pour un symbole relativement petit et asymétrique.
- Le codage arithmétique code un message entier sous la forme d'une séquence de symboles en un seul nombre décimal.
- Exemple d'utilisation : format JPEG.



Jorma J. Rissanen

Richard C. Pasco

Codage arithmétique

- **Algorithme de codage :**

1) Initialiser un premier intervalle avec deux bornes : la borne inférieure $L_c = 0$ et la borne supérieure $H_c = 1$, La taille de l'intervalle : $taille = H_c - L_c$.

2) Cet intervalle est partitionné en $\boldsymbol{N}$ sous-intervalles $[L_k, H_k[$ en fonction des probabilités de chaque symbole x_k de la source, et sa longueur $r_k = H_k - L_k$:

$$L_k = L_c + taille \times \sum_{i=0}^{k-1} p(x_i)$$

$$H_k = L_c + taille \times \sum_{i=0}^{k} p(x_i)$$

3) Choisir le sous-intervalle correspondant au prochain symbole x_k qui apparaît dans la séquence, pour redéfinir l'intervalle initial $[L_c, H_c[$

$$L_c = L_c + taille \times L_k$$

$$H_c = L_c + taille \times H_k$$

Codage arithmétique

- **Algorithme de décodage :**

1) Initialiser un premier intervalle avec deux bornes : la borne inférieure $L_c = 0$ et la borne supérieure $H_c = 1$,

   La taille de l'intervalle : $taille = H_c - L_c$.

2) Trouver le sous-intervalle $[L_k, H_k[$ du symbole

$$L_k \leq \frac{(M_c - L_c)}{taille} \leq H_k$$

3) Obtenir le symbole x_k.

4) Mettre à jour le sous-intervalle :

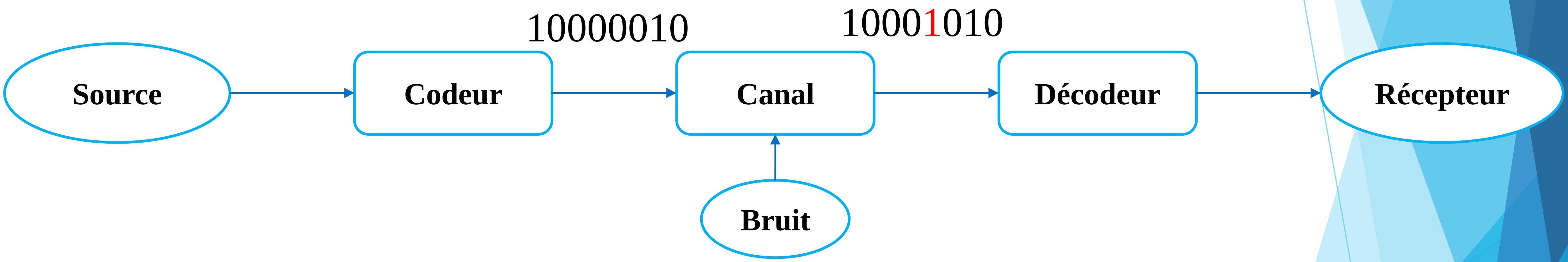
$$L_c = L_c + taille \times L_k$$

$$H_c = L_c + taille \times H_k$$

5) Répéter les étapes 2, 3, 4 et 5 jusqu’à obtenir le décodage de tous les symboles de la séquence.

Codage canal

Dans un système réel, le message reçu par le destinataire peut différer de celui qui a été émis par la source en raison de perturbations (bruit). On parle de canal bruyant.



Il faut trouver une méthode pour se protéger efficacement contre les erreurs de transmission qui affectent des symboles individuels. C’est le codage canal.

Cela se fait par ajouter des bits de la redondance au message, qui ne porteront pas d’information mais qui permettront de détecter et/ou de corriger les erreurs.

Codage canal

- Le codage canal permet de transmettre le message avec la fiabilité maximum. Plus le nombre de bit de redondance augmente, plus la correction des erreurs est meilleur. Cependant, plus le message est long (augmentation du débit binaire de la transmission) plus l'usage d'un canal coûte cher.
- Le décodeur de canal, qui connaît la loi de codage utilisée à l'émission, vient vérifier si cette loi est toujours respectée en réception. Si ce n'est pas le cas, il détecte la présence d'erreur de transmission qu'il peut corriger.
- Les méthodes de codage canal sont divisées principalement en deux modes : correction par retransmission, et autocorrection.
- Correction par retransmission ajout d'une petite quantité de redondance au message pour détecter d'éventuelles erreurs de transmission. Dans le cas d'une détection d'erreurs, le décodeur demande la retransmission du message erroné.
- Dans la autocorrection, la redondance permet de détecter et corriger au niveau du décodeur un nombre fini d'erreurs.

Codage canal

- Le code de répétition est la méthode de codage de canal la plus simple où chaque bit de message est simplement répété ***n*** fois.

- **Exemple:** Code à répétition (3,1) où chaque bit d'information est répété 3 fois

010011 → **Codage canal** → 000 111 000 000 111 111

- Les blocs codés autorisés sont 000 et 111. Le décodeur prend la décision 0 si deux ou trois 0 dans le mot, sinon prend la décision 1.

- Si le canal est très bruité, le processus de décodage peut dégrader les performances.

000 111 011 000 111 111 → **Codage canal** → 011011

- La redondance augmente le nombre de bits à transmettre (18 bits pour transmettre 6 bits) $\Longrightarrow$ problème de capacité de canal.

Codage canal

Il existe trois principaux types de codage canal : codes en bloc, codes cycliques, et codes convolutifs :

- Les codes en bloc divisent l'information en ***m*** bits et produisent un bloc de ***n*** bits codés. Ils peuvent être utilisés pour détecter ou corriger des erreurs. Les codes de bloc couramment utilisés sont les codes de Hamming.
- Les codes cycliques sont des techniques de codage linéaire par utiliser un registre à décalage pour effectuer l'encodage et le décodage (tous les mots de code sont des décalages les uns des autres).
- Les codes convolutifs sont l'un des codes de canal les plus utilisés dans les systèmes de communication pratiques. Ces codes sont développés avec une structure mathématique solide distincte et sont principalement utilisés pour la correction d'erreurs en temps réel. Il sont basés sur le codage du flux d'informations plutôt que des blocs d'informations.

Codes en bloc

Le codage en bloc s'effectue par bloc de symboles, la séquence d’information est divisée en bloc de longueur fixe ***m*** (il y ***m*** bits dans chaque séquence d’information).

Les bits de contrôle ou les bits de parité sont ajoutés à chaque bloc, où ***k*** bits sont ajoutés à chaque bloc.

Les ***k*** bits de contrôle sont calculés par le codeur en fonction des ***m*** bits d’information, en utilisant certaines règles qui déterminent le code. Les bits de contrôle sont ajoutés soit à la fin de chaque bloc, soit en modifiant complètement les blocs.

Les bits d'information et les bits de parité formant un code appelé mots-code $C(n, m)$ de longueur ***n***.

$$n = m + k$$

Chaque code de bloc est défini par le rendement (ou le taux de code) : $R = m/n$

Codes en bloc

La capacité de détection des erreurs dépend de deux paramètres, le poids de Hamming et la distance de Hamming.

Le poids de Hamming d'un mot-code est égal au nombre de bits 1 dans le mot-code.

La distance de Hamming entre deux mots-code est définie comme le nombre des positions où les deux mots diffèrent. Pour la calculer, il faut XORé, bit par bit les deux mots-code puis calculer le poids de Hamming.

La distance de Hamming minimale est définie comme la plus petite distance de Hamming existant entre les mots-codes.

La distance de Hamming minimale d'un code permet de fixer le pouvoir de détection d'un erreur, car elle définit la proximité minimale de deux mots du code et donc la capacité à les distinguer. Plus la distance entre deux mots est grande, plus il sera facile de détecter les faux mots.

Code de Hamming

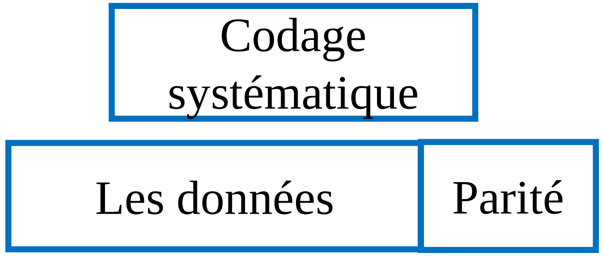
Le code de Hamming est code de bloc simple à utiliser et peut être implémenté. Ce code est utilisé pour détecter et corriger les erreurs, et il utilise le mécanisme de parité de bloc.

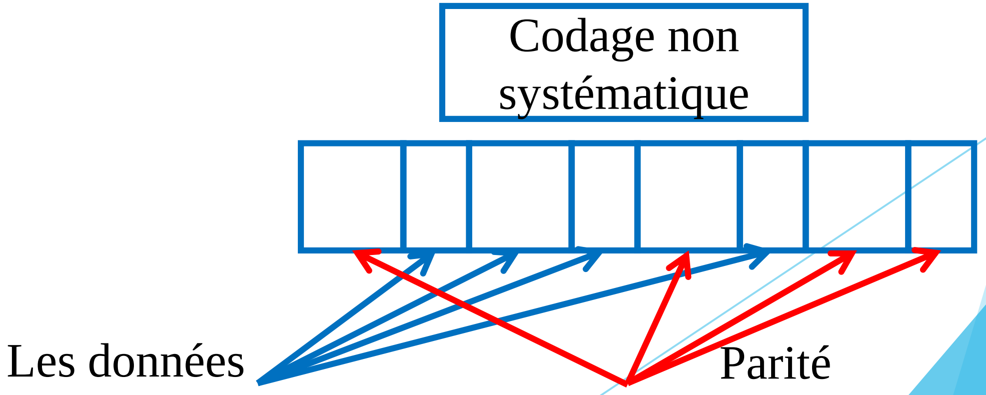
Les données sont divisées en blocs et la parité est ajoutée au bloc. Il existe deux types de code de Hamming : code systématique, et le code non systématique.

Dans le code systématique, les bits d'information se trouvent au début du mot-code, tandis que les bits de parité se trouvent à la fin du mot de code.

Dans le code non systématique, les bits d'information et les bits de parité sont mélangés dans le mot-code.

Le code Hamming standard ne peut détecter et corriger qu'une erreur d'un seul bit, et il est utilisé dans les communications spatiales et par satellite.





Code de Hamming (non systématique)

Le calcul des bits de parité est très simple, et dépendant seulement des bits d'information.

Les étapes du codage de Hamming non systématique sont :

1) Calculer le nombre de bits de parité : $2^k \geq m + k + 1.$

Avec $\boldsymbol{k}$ nombre de bits de parité, $\boldsymbol{m}$ longueur de l'information

Exemple : 10011

$$k = 2 \Longrightarrow 2^2 = 4 \geq 5 + 2 + 1 = 8, \qquad 4 \ngeq 8$$

$$k = 3 \Longrightarrow 2^3 = 8 \geq 5 + 3 + 1 = 9, \qquad 8 \ngeq 9$$

$$k = 4 \Longrightarrow 2^4 = 16 \geq 5 + 4 + 1 = 10, \qquad 16 \geq 10$$

Donc le nombre de bits de parité égale $k = 4$.

Code de Hamming (non systématique)

2) Créer le vecteur de Hamming avec longueur $m + k$.

3) Les positions de l'ordre 2^i sont utilisés pour les bits de parité, et les autres positions sont utilisées pour les bits de données.

4) Calculer les bits de parité selon :

$P_1 = P_1 + P_3 + P_5 + P_7 + \cdots,\qquad P_2 = P_2 + P_3 + P_6 + P_7 + \cdots$

$P_4 = P_4 + P_5 + P_6 + P_7 + P_{12} + \cdots,\quad P_8 = P_8 + P_9 + P_{10} + P_{11} + P_{12} + \cdots$

Exemple : 10011

9	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
1	$\boldsymbol{P_8}$	0	0	1	$\boldsymbol{P_4}$	1	$\boldsymbol{P_2}$	$\boldsymbol{P_1}$
	$\mathbf{2^3}$				$\mathbf{2^2}$		$\mathbf{2^1}$	$\mathbf{2^0}$

Code de Hamming (non systématique)

$$P_1 = P_1 + P_3 + P_5 + P_7 + P_9 = 1 + 1 + 0 + 1 = 1$$

$$P_2 = P_2 + P_3 + P_6 + P_7 = 1 + 0 + 0 = 1$$

$$P_4 = P_4 + P_5 + P_6 + P_7 = 1 + 0 + 0 = 1$$

$$P_8 = P_8 + P_9 = 1$$

9	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
1	1	0	0	1	1	1	1	1
	$\mathbf{2^3}$				$\mathbf{2^2}$		$\mathbf{2^1}$	$\mathbf{2^0}$

Le message envoyé est : 110011111.

Code de Hamming (non systématique)

Les étapes du décodage de Hamming non systématique sont :

1) Calculer les bits de parité à partir du message reçu.
2) Créer un vecteur de détection vide qui a une longueur ***k***.
3) Comparer les bits calculés avec les bits de parité dans le message reçu
4) Si les deux bits comparés sont identiques, la valeur du vecteur détecté prend 0, sinon prend 1
5) Convertir le vecteur détecté en valeur décimale pour déterminer la position du bit erroné.
6) Changer la valeur du bit erroné.

Code de Hamming (non systématique)

Exemple :

$$110011111 \longrightarrow \boxed{\textbf{Codage canal}} \longrightarrow 111011111$$

$P_1 = P_1 + P_3 + P_5 + P_7 + P_9 = 1 + 1 + 1 + 1 = 0 \qquad 0 \neq 1 \Longrightarrow 1$

$P_2 = P_2 + P_3 + P_6 + P_7 = 1 + 0 + 1 = 0 \qquad 0 \neq 1 \Longrightarrow 1$

$P_4 = P_4 + P_5 + P_6 + P_7 = 1 + 0 + 1 = 0 \qquad 0 \neq 1 \Longrightarrow 1$

$P_8 = P_8 + P_9 = 1 \qquad 1 = 1 \Longrightarrow 0$

$(0111)_2 = 7 \Longrightarrow$ bit erroné dans la position 7

$$111011111 \Longrightarrow 110011111$$

Code de Hamming (systématique)

Le calcul des bits de parité dans le code systématique dépendant une matrice appelée matrice génératrice, et dans le décodage a besoin une matrice de contrôle.

Les étapes du codage de Hamming $C(n, m)$ systématique sont :

1) Calculer le nombre de bits de parité où :

La longueur du code $n = 2^k - 1$

Le nombre de bits de parité $k = n - m$

La longueur du message $m = 2^k - k - 1$

2) Générer la matrice génératrice, puis multiplier le message binaire avec la matrice génératrice : $c = M \times G$

Où la dimension de la matrice génératrice est $m \times n$, avec une forme $G = [I_m / P_{m\times k}]$

et I_m est une matrice d'identité.

Code de Hamming (systématique)

La matrice génératrice peut être créée selon deux méthodes :

1) Génération de la matrice génératrice à partir de la matrice de contrôle ***H*** (qui génère par le côté décodage) :

- Construire la matrice de contrôle ***H*** dont les colonnes sont les nombre de 1 à ***n*** écrites sous forme binaire sur ***k*** bits.
- Permuter les colonnes de ***H*** de façon à l'écrire sous forme systématique $H = [P^T / I_k]$.
- Déterminer ***G*** par la formule : $G = [I_m / P]$

Exemple : $C(7,4)$

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \Rightarrow H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \Rightarrow G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Code de Hamming (systématique)

2) Création de la matrice génératrice directement où chaque ligne de la partie de parité prend au moins deux bits égaux à 1 (chaque vecteur n'est utilisé qu'une seule fois dans la matrice génératrice).

- La matrice de contrôle est créée à partir de la matrice génératrice à l'aide de la formule :

$$H = [P^T / I_k]$$

Exemple : $C(7,4)$

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \Rightarrow H = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Code de Hamming (systématique)

Exemple : $m = 1011 \Longrightarrow$ C (7,4)

- **Codage :** $G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$

$$c = m \times G = [1011] \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} = [1011001]$$

- **Décodage :** $d = c \times H^T$

c = $[1\textcolor{red}{0}11001] \Longrightarrow [1\textcolor{red}{1}11001]$

$$d = [1111001] \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [110]$$

$(110)_2 = 6 \Longrightarrow$ bit erroné dans la position 6 $\qquad 1\textcolor{red}{1}11001 \Longrightarrow 1\textcolor{red}{0}11001$

Codes Cycliques

Les codes cycliques sont des codes en bloc qui ont la propriété que les mots obtenus par permutations cycliques des bits d'un mot-code sont également des mots-codes.

Les codes cycliques sont utilisé dans protocole Ethernet…

Les mots d'un code cyclique sont représenté sous forme des polynômes à coefficients binaires, où représentant les bits des mots.

$$C(x) = c_{n-1}x^{n-1} + \cdots + c_2x^2 + c_1x + c_0$$

Exemple : le code 1011001 est représenté par le polynôme $C(x) = x^6 + x^4 + x^3 + 1$

Les mots-code d'un code cyclique sont générés par un polynôme générateur de degré $n - m$ (de degré k) , ce polynôme représenté par : $G(x) = x^k + g_{k-1}x^{k-1} + \cdots + g_1x + 1$. Le polynôme générateur doit être primitif, c'est-à-dire :

- Le polynôme doit être diviseur de $x^n + 1$, avec $n = 2^k - 1$.
- Le polynôme doit être irréductible.

Codes Cycliques

La matrice génératrice peut être obtenu à partir du polynôme générateur où le degré dans cette matrice est égale à ***n*** :

$$G = \begin{bmatrix} x^{m-1}G(x) \\ x^{m-2}G(x) \\ \vdots \\ xG(x) \\ G(x) \end{bmatrix}$$

La relation entre le polynôme générateur et le polynôme de contrôle est donnée par : $G(x) \times H(x) = x^n + 1$

La matrice de contrôle peut être obtenu à partir du polynôme de contrôle où le degré dans cette matrice est égale à ***n*** :

$$H = \begin{bmatrix} H(x)/x^{k-1} \\ H(x)/x^{k-2} \\ \vdots \\ H(x)/x \\ H(x) \end{bmatrix}$$

Avec $G \times H^T = 0$.

Remarque : Les lignes de la matrice ***H*** représentent les polynômes écrits de gauche à droite, en commençant par le poids le plus faible.

Codes Cycliques

Le mot-code $C(x)$ du message $M(x)$ est obtenu par : $C(x) = M(x) \times G(x)$

Exemple : $M = [1011]$ et $G(x) = x^3 + x^2 + 1$

$$C(x) = M(x) \times G(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$

Pour décoder le mot-code $C(x)$ (détecter si il y a une erreur et obtenir le message $M(x)$), il faut :

- Calculer le reste de la division du mot reçu par le polynôme génératrice $G(x)$:

$$S(x) = reste(\frac{C(x)}{G(x)})$$

Où le polynôme $S(x)$ est de degré $\boldsymbol{k-1}$.

- Si $S(x) = 0$, prendre comme message le résultat de la division.
- Si $S(x) \neq 0$, Vérifier le tableau des syndromes pour détecter l'erreur et corrigez-la. Puis, diviser le polynôme corrigé avec le polynôme générateur pour obtenir le message.

Codes Cycliques

Exemple : $M = [1011]$ et $G(x) = x^3 + x^2 + 1$

$$C(x) = M(x) \times G(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$

$C(x) = M(x) \times G(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 \Rightarrow C'(x) = x^6 + x^5 + x^4 + x^3 + x + 1$

Dans le décodage :

$$S(x) = \frac{C'(x)}{G(x)} = x^2 = [100]$$

D'après le tableau des syndromes, l'erreur est dans la position 3.

$C = [1111011] \Rightarrow [1111111]$

$\Rightarrow C(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

$$\frac{C(x)}{G(x)} = x^3 + x + 1 \Rightarrow M(x) = x^3 + x + 1 \Rightarrow M = [1011]$$

Position de l'erreur	$\boldsymbol{S(x)}$
0 0 0 0 0 0 0	**0 0 0**
0 0 0 0 0 0 1	**0 0 1**
0 0 0 0 0 1 0	**0 1 0**
0 0 0 0 1 0 0	**1 0 0**
0 0 0 1 0 0 0	**1 0 1**
0 0 1 0 0 0 0	**1 1 1**
0 1 0 0 0 0 0	**0 1 1**
1 0 0 0 0 0 0	**1 1 0**

Codes Cycliques

Les codes cycliques peuvent construire les codes systématiques, pour cela il faut :

- Multiplier le polynôme du message par x^k: $M(x) \times x^k$.
- Diviser cette multiplication par le polynôme génératrice $G(x)$: $M(x) \times x^k / G(x)$.
- Ajouter le reste de cette division à la multiplication pour obtenir le mot-code $C(x)$:

$$C(x) = \left(M(x) \times x^k\right) + r(x).$$

Exemple : $M = [1011]$ et $G(x) = x^3 + x^2 + 1$ et $k = 3$

$$M(x) \times x^3 = x^6 + x^4 + x^3$$

$$r(x) = x^2$$

$$C(x) = x^6 + x^4 + x^3 + x^2$$

Remarque : Le décodage suit les mêmes étapes que le décodage non symétrique.