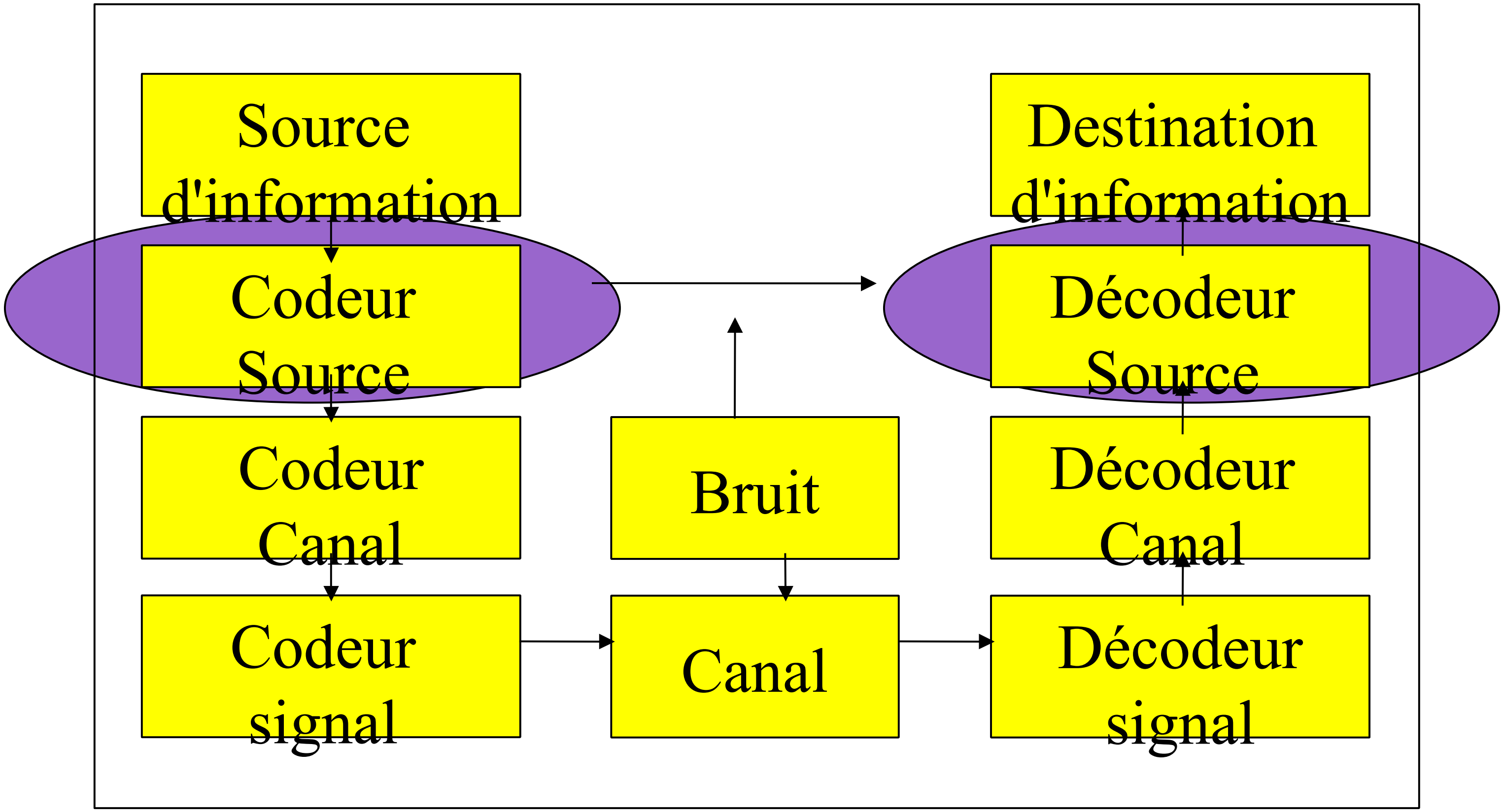
THEORIE DE L'INFORMATION

chapitre 2

Codage source

jp.guédon **INFO 3**

Théorie de l' Information



2. CODAGE ENTROPIQUE DE L'INFORMATION

2.1. Définitions relatives au codage

- 2.1.1 Codage et décodage
- 2.1.2 Débit
- 2.1.3 Efficacité
- 2.1.4 Redondance
- 2.1.5 Exemples

2.2. Algorithmes de codage entropique

2. CODAGE ENTROPIQUE DE L'INFORMATION

Le codage de l'information consiste à exprimer les messages d'une source grâce à un alphabet particulier. L'objectif est multiple :

- \- **transcrire l'information** pour une meilleure manipulation en fonction des applications envisagées. Exemple du numéro INSEE pour les applications médicales.
- \- **réduire la quantité** de symboles à transmettre

2. CODAGE ENTROPIQUE DE L'INFORMATION

- \- **réduire la quantité** de symboles à transmettre
- \- **prévenir les dégradations** (distorsions et bruits du canal) en optimisant la décision sur un symbole à la destination
- \- **sécuriser** la transmission des informations (secret) pour la rendre inintelligible excepté au destinataire possédant le code

- **Codage** : soit un alphabet A formé de q caractères A = {a1, a2, ..., aq} et un ensemble de messages d'une source S = {m1, m2, ..., mN}.

  Un code est une application de S dans A.

- L'élément Mi de A qui correspond à mi est appelé le mot-code de mi.
- Sa longueur li est le nombre de caractères qui le forment.
- Une suite de messages est appelé un texte.

- **Exemple**:

S = {m1,m2} , A = {0,1}, on forme deux codes :

C1 ={ m1 →0; m2 →1}, C2 = {m1→ 0, m2 →10}

Messages	m1	m2
C1	0	1

Messages	m1m1	m1m2	m2m1	m2m2
C2	00	010	100	1010

- **Codage en blocs**:

un code en blocs est aussi une application mais de S^k dans A.

La différence est que l'on affecte à une suite de k messages un seul mot-code.

Exemple : suite binaire S = {0,1} A = entiers

0000 11 00000 1111 0 11 => 4 2 5 4 1 2

10000 11 00000 1111 0 11 => 0 1 4 2 5 4 1 2

2.1.2 Débit

Comme on peut le voir dans l'exemple précédent le nombre de caractères d'un mot-code dépend de celui-ci. On veut maintenant caractériser l'information moyenne transportée par un codage.

li = long(Mi) est le nombre de caractères de A utiles pour le codage de mi.

Si la source S utilise N messages différents, la longueur moyenne est calculée par :

2.1.2 Débit

li = long(Mi) est le nombre de caractères de A utiles pour le codage de mi avec pi = Pr{m=mi},

si la source S utilise N messages différents, la longueur moyenne est calculée par :

$$L = \sum_{i=1}^{N} p(a_i)\, l_i \, bit/symbole$$

exercice

2.1.2 Débit

source	a1	a2	a3	a4	a5	a6
prob	1/2	1/8	1/8	1/8	1/16	1/16
Code 1	000	001	010	011	100	101
Code 2	0	100	101	110	1110	1111

Calculer l'entropie de la source avec les 6 symboles

Calculer la longueur moyenne de chacun des 2 codes solution

$$H(A) = \sum_{i=1}^{6} p(a_i) \log_2\left(\frac{1}{p(a_i)}\right) bit/symbole$$

$$H(A) = \frac{1}{2}\log_2(2) + 3\left(\frac{1}{8}\log_2(8)\right) + 2\left(\frac{1}{16}\log_2(16)\right) bit/symbole$$

$$L_1 = \sum_{i=1}^{N} p(a_i)\, l_i \; bit/symbole$$

solution

$$L_1 = \frac{1}{2} \times 3 + 3\left(\frac{1}{8} \times 3\right) + 2\left(\frac{1}{16} \times 3\right) bit/symbole$$

$$L_2 = \frac{1}{2} \times 1 + 3\left(\frac{1}{8} \times 3\right) + 2\left(\frac{1}{16} \times 4\right) bit/symbole$$

$$L_1 = \sum_{i=1}^{N} p(a_i)\, l_i \; bit/symbole$$

solution

$$H(A) = 2 + \frac{1}{8} \; bit/symbole$$

$$L_1 = 3 \; bit/symbole$$

$$L_2 = 2 + \frac{1}{8} \; bit/symbole$$

H et L s’expriment
dans la même unité
(sont comparables)

H ne dépend pas du code
H ≤ L

2.1.2 Débit

L'entropie par caractère est donc égale à

$$\frac{H(S)}{L}\, bit / caractère$$

On se rappelle aussi que l'entropie est bornée et puisqu'on a q caractères dans A :

$$0 \leqslant \frac{H(S)}{L} \leqslant \log_2 q$$

2.1.2 Débit

La longueur moyenne par caractère L est donc BORNEE par :

$$\boxed{\frac{H(S)}{\log_2 q} \leqslant L}$$

D'autre part, si la source produit D caractères par seconde, le débit d'information exprimé en bit/ seconde est :

$$R = D.\frac{H(S)}{L}$$

Remarques

1) plus L est petit (H étant donné par la source), plus le débit d'information est important

2) la limitation minimum sur L induit :

$$R \leqslant D.log_2 q$$

2.1.4 Redondance

La redondance est utilisée pour accroître la robustesse aux erreurs de transmission des codes (cf. théorie des codes détecteurs / correcteurs). Elle est définie par :

$$\boxed{\mathrm{red} = 1 - \mathit{eff}} = 1 - \frac{H(S)}{L.log_2(q)}$$

Donc plus un code est efficace (en termes de transmission) moins il est redondant (robuste).

2.1.4 Redondance

Ou est le compromis entre efficacité et redondance ?

Il dépend essentiellement du médium de transmission et du taux d'erreurs binaires de celui-ci et aussi du ou des protocoles supportés au-dessus.

Des valeurs classiques de redondance utilisées au niveau du codage canal radioélectrique sont 1□ 2, 2□ 3, 3□ 4 ce qui veut dire 2 bits envoyés pour 1 crée par le codeur source (ou bien 3 pour 2, ou 4 pour 3).

2.1.3 Efficacité

Pour un alphabet A associé à une source S de code C, l'efficacité est le rapport :

$$eff = \frac{min\,L}{L}$$

Ce minimum est donné par : $min\,L = \frac{H(S)}{\log_2(q)}$

ce qui fait

$$eff = \frac{H(S)}{L.log_2(q)}$$

Remarque : par définition $0 \leqslant eff \leqslant 1$

2.1.5 Exercice

La source S produit 4 sortes de messages de probabilités respectives telles qu'indiquées dans le tableau suivant:

Calculer l'entropie de cette source.

Messages	m1	m2	m3	m4
proba	1/2	1/4	1/8	1/8
C1	00	01	10	11
C2	0	10	110	111

pour les deux codes C1 et C2, calculer la longueur moyenne et en déduire l'efficacité et la redondance.

2.1.5 Exemples

La source S produit 4 sortes de messages de probabilités respectives telles qu'indiquées dans le tableau suivant:

$H(S) = ½ \log2(2) + 1/4 \log2(4) + 2.\ 1/8 \log2(8) = 1.75$ bit/symbole

Messages	m1	m2	m3	m4
proba	1/2	1/4	1/8	1/8
C1	00	01	10	11
C2	0	10	110	111

2.1.5 Exemples

La source S produit 4 messages

$$H(S) = 1.75 \text{ bit/symbole}$$

Messages	m1	m2	m3	m4
proba	1/2	1/4	1/8	1/8
C1	00	01	10	11
C2	0	10	110	111

$$eff_1 = \frac{H(S)}{L.log_2(q)} = \frac{1.75}{2} = 87.5\% \qquad \text{red}_1 = 1 - eff_1 = 22.5\%$$

$$eff_2 = \frac{H(S)}{L.log_2(q)} = \frac{1.75}{1.75} = 100\% \qquad \text{red}_2 = 1 - eff_0 = 0\%$$

2.1.5 Autres exemples

La source S produit 4 messages H(S) = 1.75

Messages	m1	m2	m3	m4
proba	1/2	1/4	1/8	1/8
C3	00	00	10	11
C4	0	1	11	01

C3 n'est pas un code régulier (m1 et m2 même codage).
C4 est régulier mais peut être ambigu à la lecture :
111 => m2m3 ou m3m2, 01 => m1m2 ou m4 ?

2.2. Codage de longueur fixe

Toute personne résident en France possède un code sur 13 caractères ce qui est finalement peu pour nous résumer!

L'immense avantage qu'en tire tout traitement est évident. Cette opération est très générale dans tout secteur de traitement de données textuelles, images, vidéo ou sonores.

Le plus connu : code ASCII

Utile pour les couches basses des réseaux (plus de sémantique) => utilisation de la redondance.

2.2. Codage de longueur variable

S'il fallait que tous les mots du dictionnaire possèdent le même nombre de lettres on n'en finirai pas et cela prendrait sans doute pas mal de place! Il y a donc des cas de figure ou il vaut mieux procéder par des code de longueur variable.

Le principe est alors simple le mot code qui revient le plus souvent doit pouvoir être codé avec le plus faible nombre de bits alors qu'il sera possible d'utiliser plus de bits pour un mot rare (au sens de l'entropie).

2.2. Codage de longueur variable

2.2.1. Principe

À partir d'un ensemble de mots et de leurs probabilités respectives on veut associer des codes binaires dont la longueur sera inversement proportionnelle à la probabilité.

On veut donc trouver un codage de

A={ a1, a2,...,aS} et Proba={p(a1),...,p(aS)}

vers Code={c1,c2,...,cS} et Long={l1,l2,...,lS}

Le décodage est l'opération inverse de A vers Mots

La longueur du code est alors:

$$L = \sum_{i=1,S} p(a_i)\, l_i \; bit/symbole$$

2.2. Codage de longueur variable

Théorème Shannon (1948) :

Pour des codes binaires, on peut approcher L aussi proche que l'on veut par H(A)

(En fait on a : H(A) ≤ L ≤ H(A)+1)

$$L = \sum_{i=1,S} p(a_i) l_i \; bit/symbole$$

$$H(A) = E\{I_N(A)\} = -\sum_{i=1,S} p(a_i)\log_2(p(a_i)) \; bit/symbole$$

2.2. Codage de longueur variable

Théorème:

Pour des codes binaires, on peut approcher L aussi proche que l'on veut par H(A)

Donc il suffit de trouver un algorithme de codage optimal

(le théorème ne dit pas comment faire pour trouver cet algorithme…)

2.2. Codage de longueur variable:

A partir de maintenant, nous partons à la pêche au bon algorithme de codage

Pour comparer, on prend la source A suivante:

ai	1	2	3	4	5	6	7
p(ai)	0.44	0.19	0.18	0.07	0.07	0.03	0.02

Son entropie est égale à H= ?

$$H(A) = E\{I_N(A)\} = -\sum_{i=1,S} p(a_i)\log_2(p(a_i))\, bit/symbole$$

2.2. Codage de longueur variable:

$$H(A) = E\{I_N(A)\} = -\sum_{i=1,S} p(a_i)\log_2(p(a_i))\, bit/symbole$$

Pour la source A

H(A) = - {0,44 log2(0,44) +0 ,19 log2(0,19) +0,18 log2(0,18) +2x0,07 log2(0,07) +0,03 log2(0,03)+0,02 log2(0,02) } bit/symbole

ai	1	2	3	4	5	6	7
p(ai)	0.44	0.19	0.18	0.07	0.07	0.03	0.02

2.2. Codage de longueur variable:

Pour la source A

H(A) = - {0,44 log2(0,44) +0 ,19 log2(0,19) +0,18 log2(0,18) +2x0,07 log2(0,07) +0,03 log2(0,03)+0,02 log2(0,02) }bit/symbole

En utilisant log2(x)=loga(x)/loga(2) (ici a=e)

on a H(A) = (-1/ln(2)) {0,44 ln(0,44) +0,19 ln(0,19) +0,18 ln(0,18) +2x0,07 ln(0,07) +0,03 ln(0,03)+0,02 ln(0,02) } bit/symbole

2.2. Codage de longueur variable:

Pour la source A

on a $H(A) = (-1/\ln(2)) \{0{,}44 \ln(0{,}44) +0{,}19 \ln(0{,}19) +0{,}18 \ln(0{,}18) +2\text{x}0{,}07 \ln(0{,}07) +0{,}03 \ln(0{,}03)+0{,}02 \ln(0{,}02) \}$

$H(A) = (1/\ln(2)) \{0{,}361-0{,}19 \ln(0{,}19) -0{,}18 \ln(0{,}18) -2\text{x}0{,}07 \ln(0{,}07) -0{,}03 \ln(0{,}03)-0{,}02 \ln(0{,}02) \}$

$H(A) = (1{,}443) \{0{,}361+0{,}3155 + 0{,}3087 + 2\text{x}0{,}186 +0{,}105 +0{,}078 \}$

$H(A) = (1{,}443)\text{x}1{,}5402 = 2{,}223$ bit/symbole

2.2. Codage de longueur variable:

- Codage à virgule

De la même façon qu'une virgule sépare deux propositions, ici la "virgule" va séparer deux mots : c'est ainsi que l'on se rends compte que le premier est fini et que le second débute.

Le but est alors de rester plus robuste à l'erreur binaire qui intervient si souvent... Ceci simplement parce que même si l'on manque une virgule le reste du texte est encore compréhensible (on a perdu deux mots).

2.2. Codage de longueur variable:

- Codage à virgule

On met en œuvre en codant par exemple « 0 » pour la virgule

« 0 » pour le code de plus grande probabilité

« 10 » pour le suivant

« 110 » pour le suivant

Etc.

2.2. Codage de longueur variable: Codage à virgule

Exemple:

ai	p(ai)	code(ai)	ai	p(ai)	code(ai)
1	0.44	0	5	0.07	11110
2	0.19	10	6	0.03	111110
3	0.18	110	7	0.03	1111110
4	0.07	1110			

2.2. Codage de longueur variable: Codage à virgule

Exemple source caractérisée par son entropie H=2,223 :

ai	p(ai)	code(ai)	ai	p(ai)	code(ai)
1	0.44	0	5	0.07	11110
2	0.19	10	6	0.03	111110
3	0.18	110	7	0.03	1111110
4	0.07	1110			

Avec ce codage, on obtient une longueur moyenne L=0,44+0,19x2+0,18x3+0,07(4+5)+0,03x6+0,02x7

L= 0,44+0,38+0, 54 +0,63 +0,18+0,14

L= 0,82 +1,17 +0,32

L = 2,31 bit/symbole

2.2. Codage de longueur variable: Codage à virgule comparaison avec codage longueur fixe

ai	p(ai)	code(ai)	ai	p(ai)	code(ai)
1	0.44	0	5	0.07	11110
2	0.19	10	6	0.03	111110
3	0.18	110	7	0.03	1111110
4	0.07	1110			

Avec ce codage, on obtient une longueur moyenne :

Lfixe =3x(0,44+0,19+0,18+0,07+0.07+0,03+0,02)

Lfixe= 3x1 =3

à comparer avec L = 2,31 pour le code à virgule

2.2. Codage de longueur variable:

- Codage à virgule

Exemple

Pour la source caractérisée par son entropie
H=2,223 bit/symbole
et par un codage virgule avec L = 2,31 bit/symbole

Rappel du théorème de Shannon :
H=2,223 <= L=2,31 <= 3,223=H+1

On est déjà proche de la limite!

2.2. Codage de longueur variable: Codage virgule « G »

Dans le code à virgule, pas très efficace sur la longueur des li: on modifie l'algo pour ne pas avoir de code avec deux « 1 » à suivre, on commence un code par un « 1 » et on met un « 1 » pour faire la virgule

Les codes vont donc être construit par la suite des codes ordonnés par proba:

s1	s2	s3	s4	s5	s6	s7	s8
1	101	100 1	100 01	101 01	100 001	100 101	101 001

2.2. Codage de longueur variable:

Codage virgule « G »

Codage: soit la suite :s1s2s1s3s1s1s2s4…

On obtient la suite :1101110011110110101…

Décodage:

À chaque fois que l'on a 2 « 1 » à la suite, le premier est la fin du code précédent et le second commence le suivant:

1□ 101 □ 1 □ 1001 □ 1 □ 1 □ 101 □ 10101

s1s2s1s3s1s1s2s4

2.2. Codage de longueur variable: Codage virgule « G »

Comparaison virgule « G » avec virgule simple: on perd dans les premiers codes (s2,s3, et s4) mais on gagne ensuite pour tous les autres.

1	101	1001	10001	10101	100001
1	01	001	0001	00001	000001
100101	101001	1000001	1000101	1001001	1010001
0000001	00000001	000000001	0000000001	00000000001	000000000001

2.2. Codage de longueur variable: Codage virgule « G »

Les codes virgule « G »:

1 101 1001 10001 10101 100001 100101 101001

Par construction, on ajoute les codes (sans la virgule) à partir d'une puissance de 2 jusqu'à la puissance de deux inférieure

1 10 100 1000

On ajoute à 10000, 10 puis 100

(ni le premier ni le dernier)

2.2. Codage de longueur variable: Codage virgule « G »

Exemple:

ai	p(ai)	code(ai)	ai	p(ai)	code(ai)
1	0.44	1	5	0.07	10101
2	0.19	101	6	0.03	100001
3	0.18	1001	7	0.03	100101
4	0.07	10001			

2.2. Codage de longueur variable:

Avec ce codage, on obtient une longueur moyenne
L=0,44+0,19x3+0,18x4+0,07(5+5)+0,03x6+0,02x6

L= 0,44+0,57+0, 72 +0,70 +0,18+0,12

L= 1,01 +1,42 +0,30 => L = 2,73 bit/symbole

H=2,223 <= L(virgule)=2,31 <L(virgule « G »)=2,73

2.2. Codage de longueur variable:

Codage de Shannon-Fano (1949)

Comme on a vu : il faut mettre le moins de longueur sur les codes de plus grandes probas:

On repense à la définition de l'entropie : idée pour associer les codes dans un arbre binaire avec le parcours de la racine à la feuille.

La première façon de construire cet arbre est de partitionner les symboles en deux de sorte à avoir deux sous arbres équilibrés (dichotomie)

2.2. Codage de longueur variable: Codage de Shannon-Fano

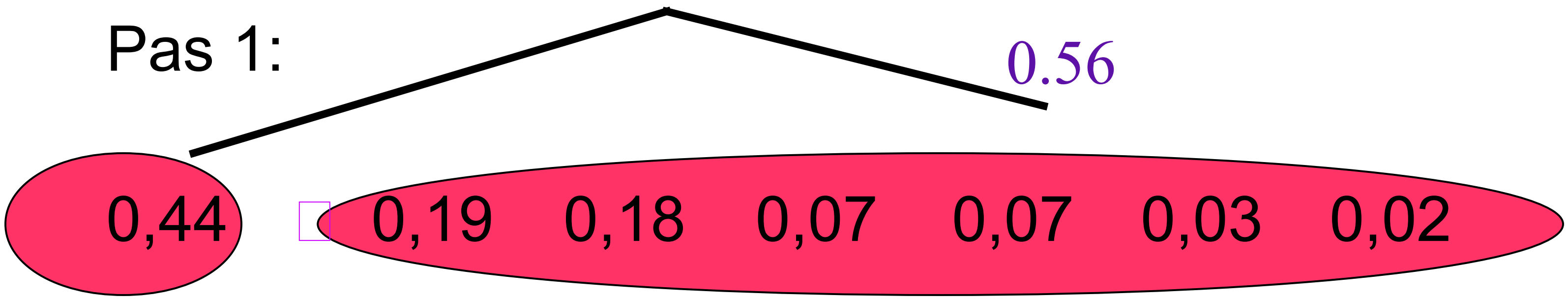
La première façon de construire cet arbre est de partitionner les symboles en deux de sorte à avoir deux sous arbres équilibrés

Exemple favori:

ai	1	2	3	4	5	6	7
p(ai)	0.44	0.19	0.18	0.07	0.07	0.03	0.02

2.2. Codage de longueur variable: Codage de Shannon-Fano

Pas 1:



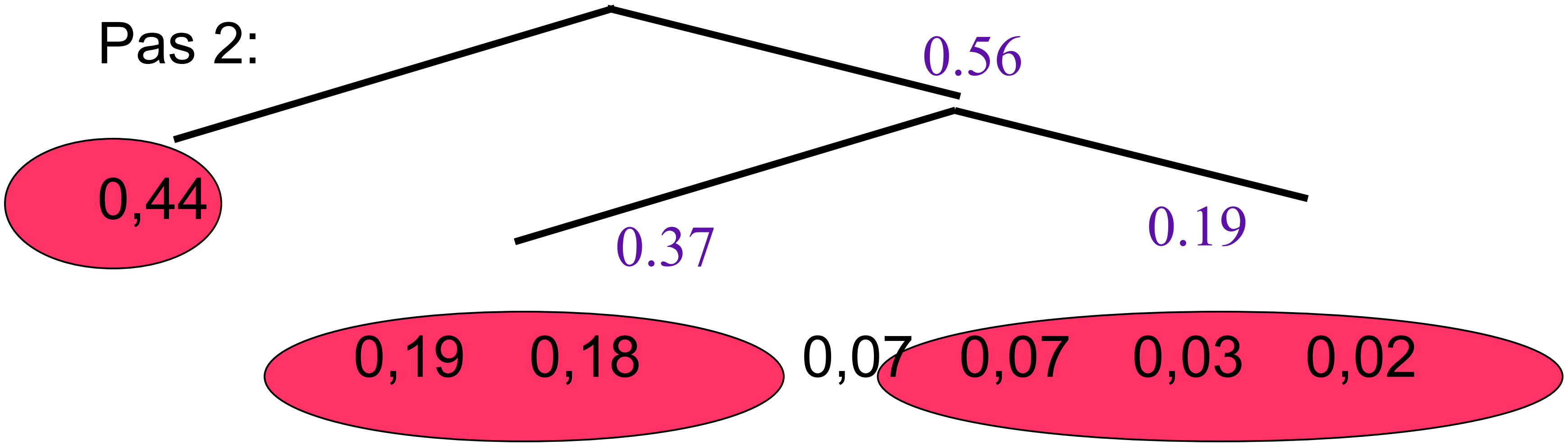
on sépare ici car le mieux équilibré

(0,44 □ 0,56) sinon (0,44+0,19 □ 056-0,19)

Mais il faut aussi laisser l'ordre de décroissance des probabilités dans l'arbre (pour savoir où on a rangé les valeurs)

2.2. Codage de longueur variable: Codage de Shannon-Fano

Pas 2:



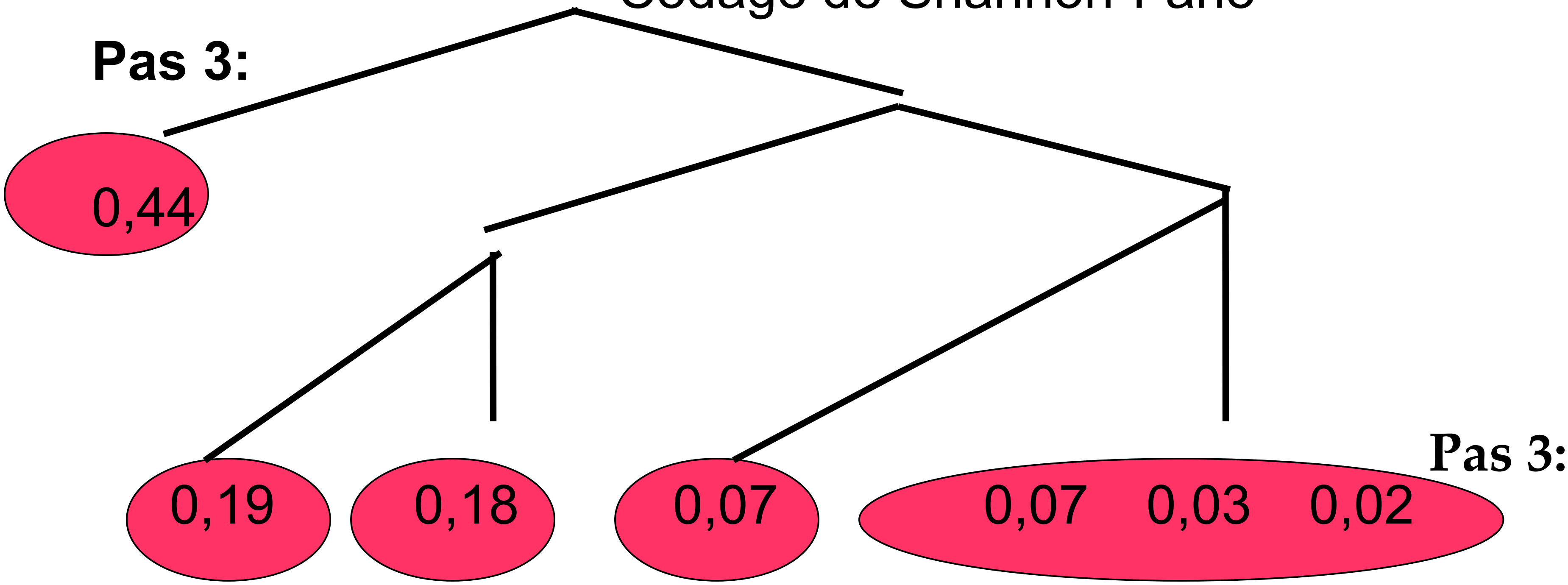
56/2=28 soit 19 soit 37 donc pareil : 28-19=37-28

On choisit 37 car mieux équilibré (nb de feuilles)

2.2. Codage de longueur variable:

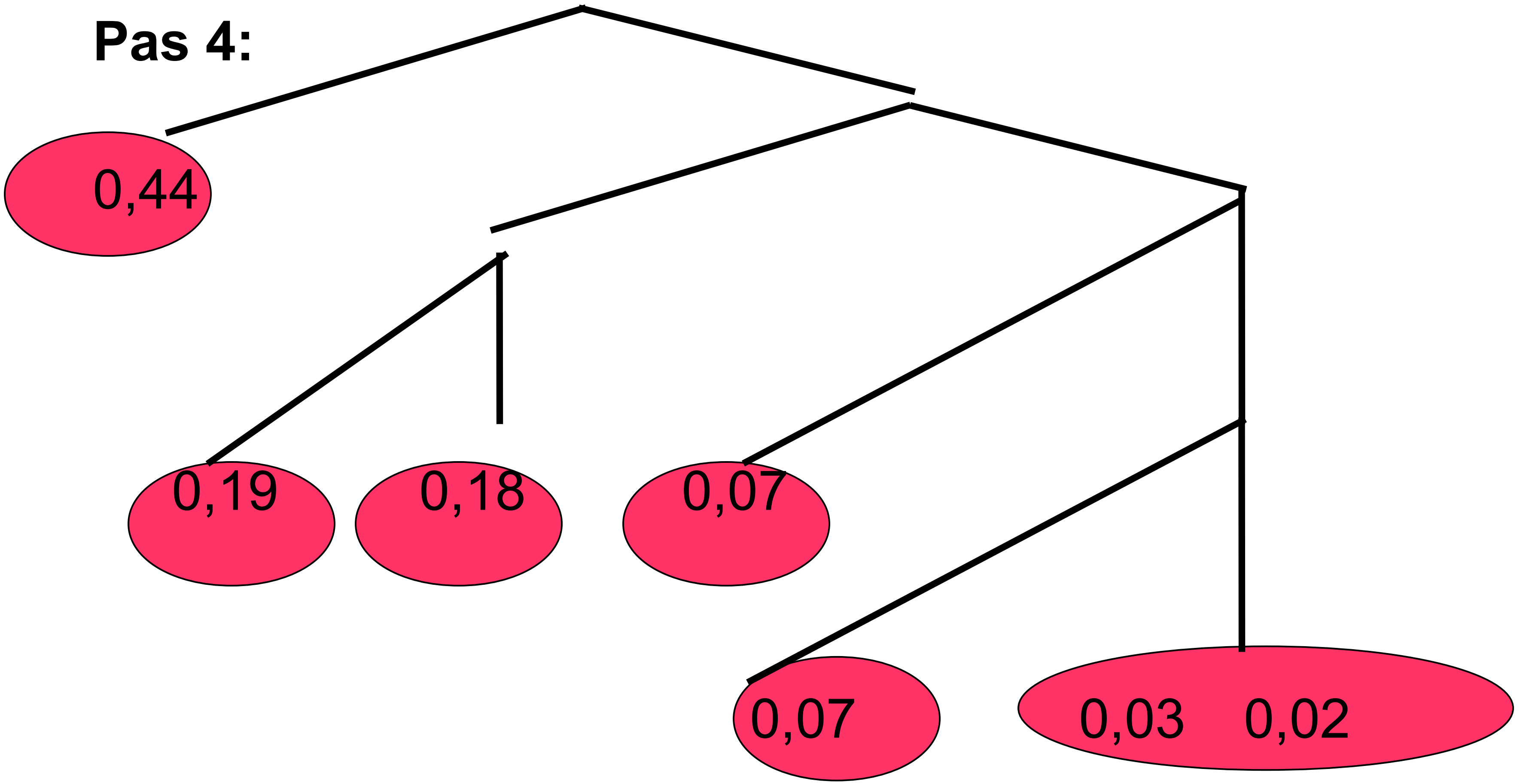
Codage de Shannon-Fano

Pas 3:



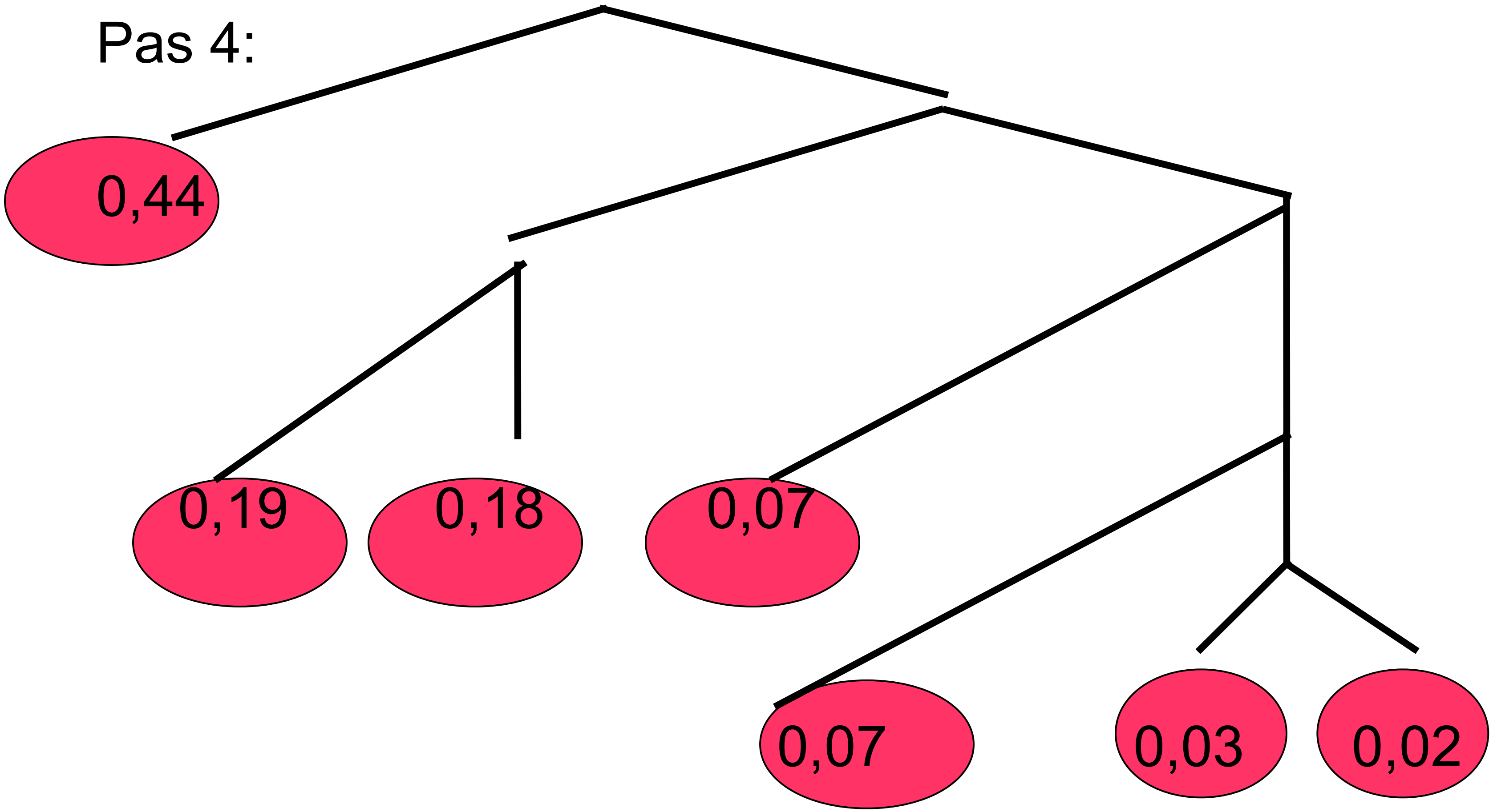
2.2. Codage **de longueur variable: Codage de Shannon-Fano**

Pas 4:

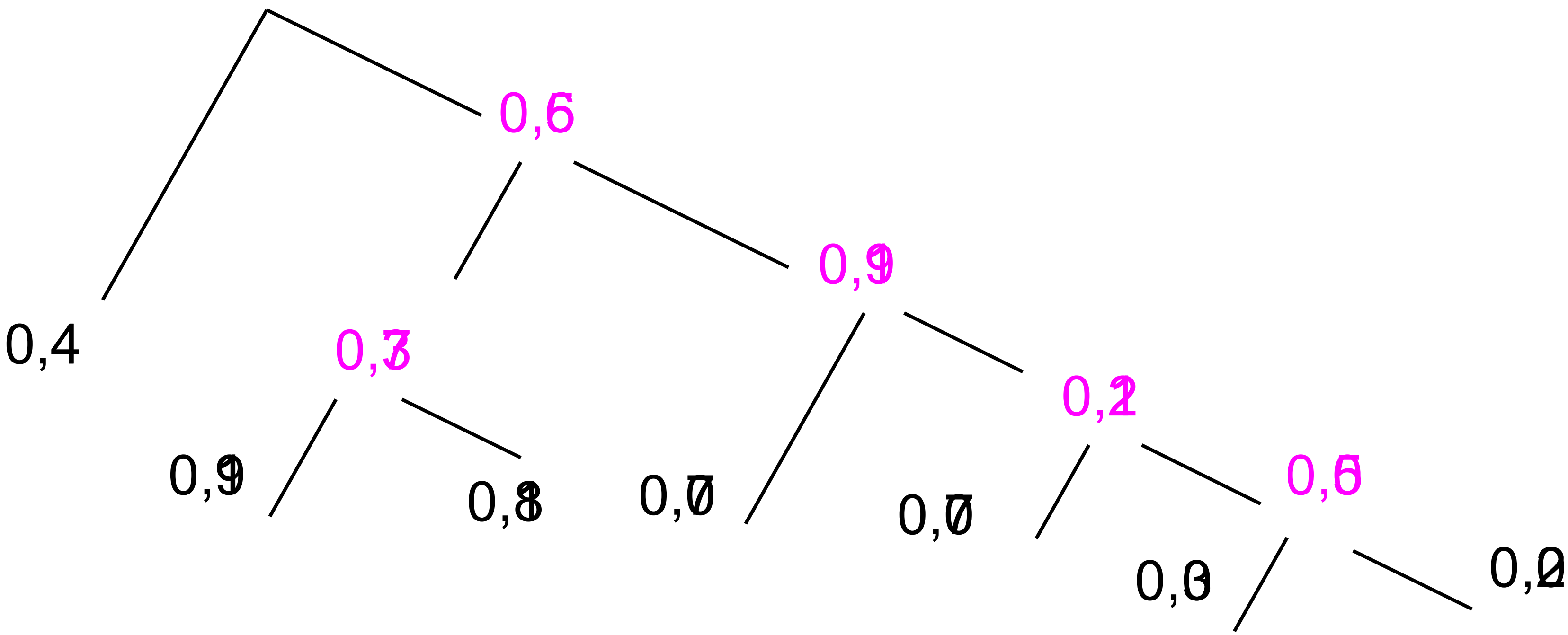


2.2. Codage de longueur variable: Codage de Shannon-Fano

Pas 4:

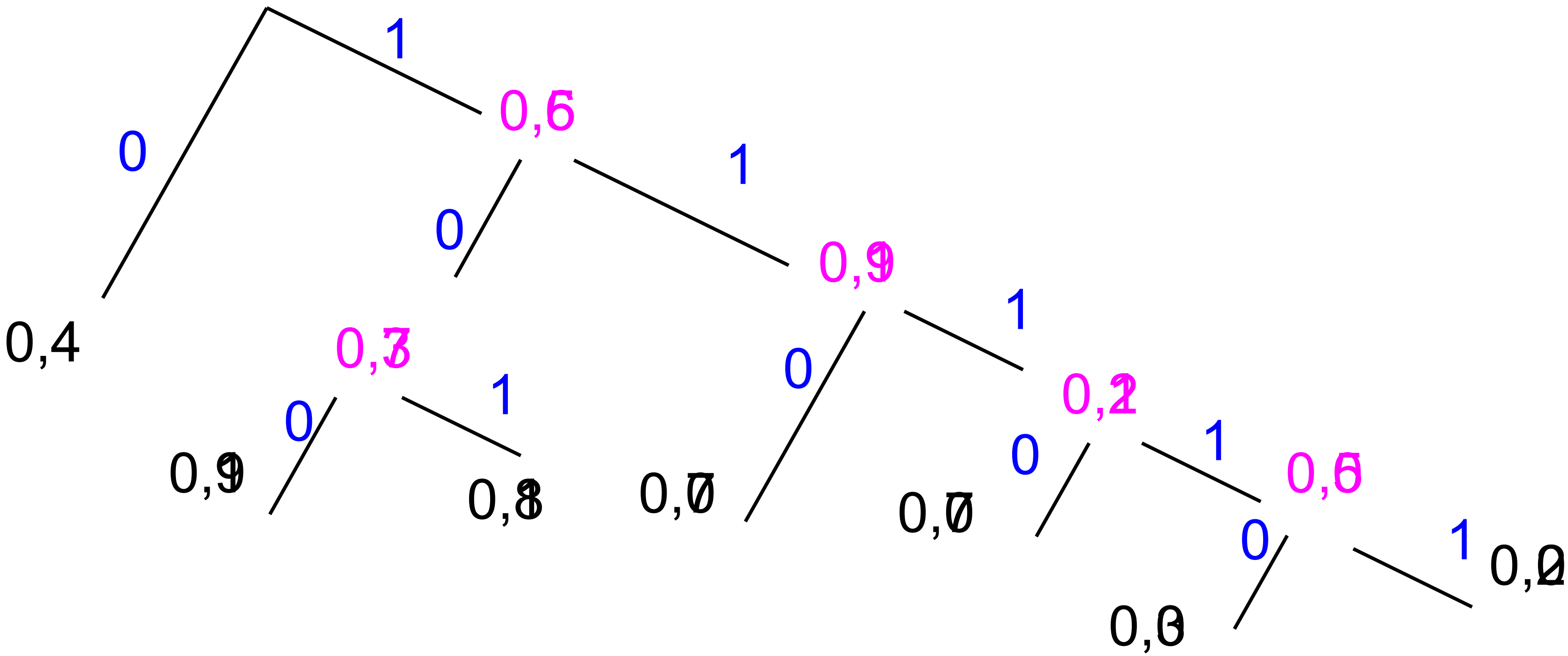


2.2. Codage de longueur variable: Codage de Shannon-Fano



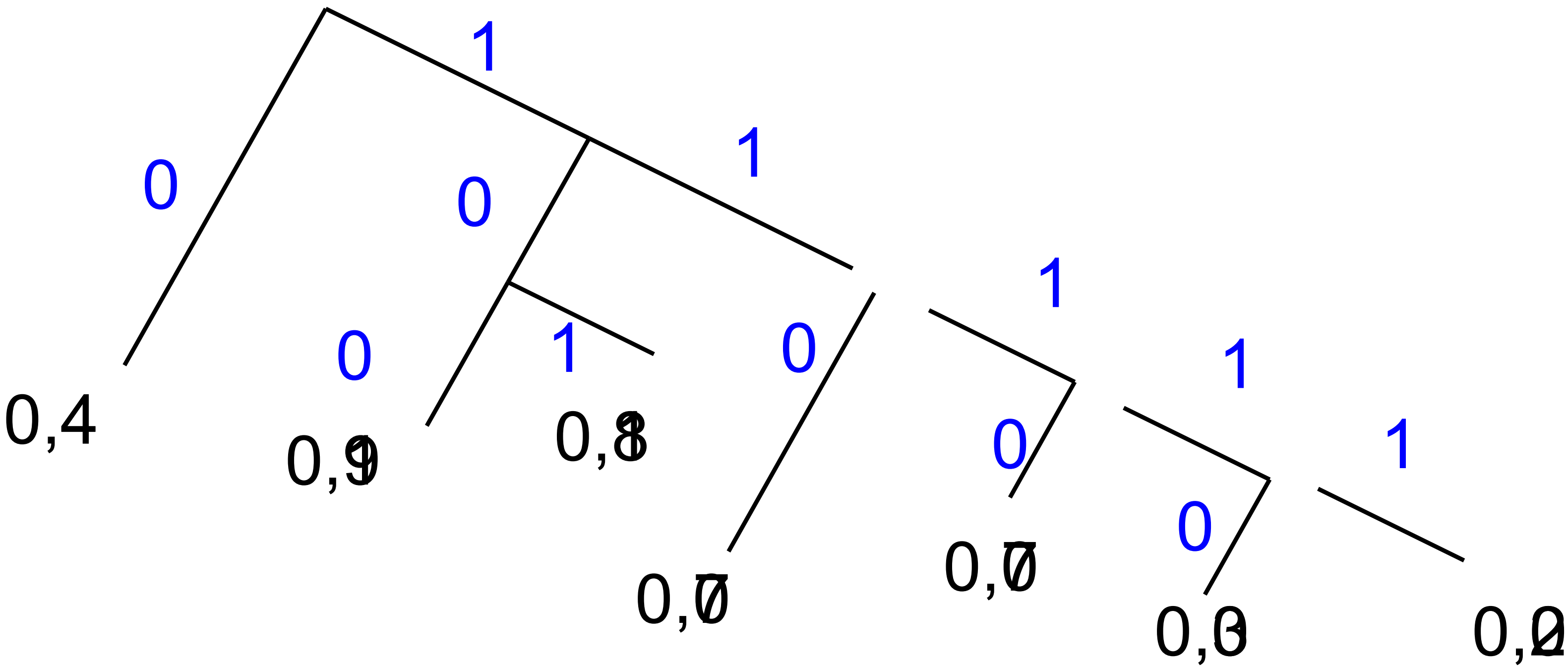
2.2. Codage de longueur variable:

Codage de Shannon-Fano



2.2. Codage de longueur variable:

Codage de Shannon-Fano

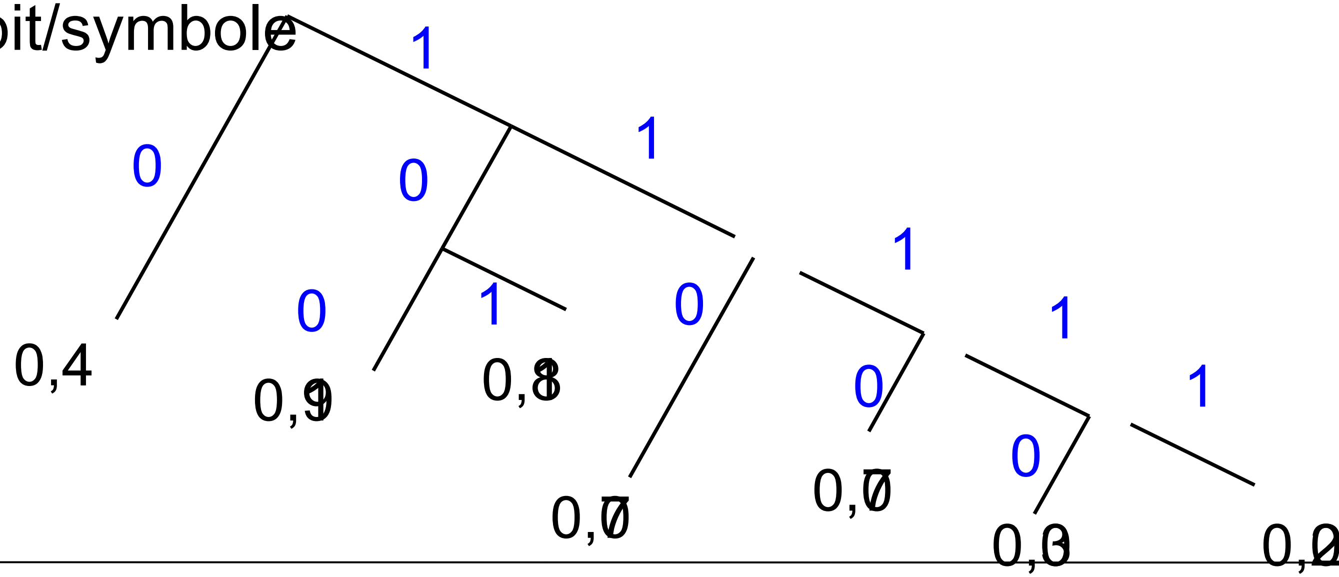


2.2. Codage de longueur variable: Codage de Shannon-Fano

PROB	**0,18**	0,18	0,07	0,07	0,03	0,02	0,02
CODE	1	100	101	110	1110	11110	11111

L= 0,44+3x0,19 +3x0,18 +3x0,07+4x0,07+5x0,03 +5x0,02

L= 2,23 bit/symbole



2.2. Codage de longueur variable:

H=2,223 <= L(Shannon - Fano)=2,23

<=L(virgule)=2,31

<(virgule « G »)=2,73

2.2. Codage de longueur variable:

Codage de Huffman (1951)

Même idée que Shannon Fano du codage entropique de l'arbre binaire avec le parcours de la racine à la feuille.

La différence : on part des feuilles que l'on agrège pour aller vers la racine.

(il faut aussi laisser l'ordre de décroissance des probabilités dans l'arbre)

2.2. Codage de longueur variable:

Codage de Huffman

Exemple favori:

ai	1	2	3	4	5	6	7
p(ai)	0.44	0.19	0.18	0.07	0.07	0.03	0.02

A chaque étape on agrège dans un nœud les deux plus petites probabilités

2.2. Codage de longueur variable:

Codage de Huffman

Exemple favori:

6+7 p=0,05

5+6+7 p=0,12

4+5+6+7 p=0,19

Ici on a 2 nœuds avec 0,19, un avec 0,18 et un 0,44 : on peut donc agréger le 0,19 de notre choix avec le 0,18

ai	1	2	3	4	5	6	7
p(ai)	0.44	0.19	0.18	0.07	0.07	0.03	0.02

2.2. Codage de longueur variable:

Codage de Huffman

Exemple favori:

6+7 p=0,05. 5+6+7 p=0,12. 4+5+6+7 p=0,19

Ici on a 2 nœuds avec 0,19, un avec 0,18 et un 0,44 : on peut donc agréger le 0,19 de notre choix avec le 0,18

ai	1	2	3	4	5	6	7
p(ai)	0.44	0.19	0.18	0.07	0.07	0.03	0.02

2.2. Codage de longueur variable:

Codage de Huffman

Exemple favori:

6+7 p=0,05. 5+6+7 p=0,12. 4+5+6+7 p=0,19

(Choix identique à exemple Shannon-Fano)

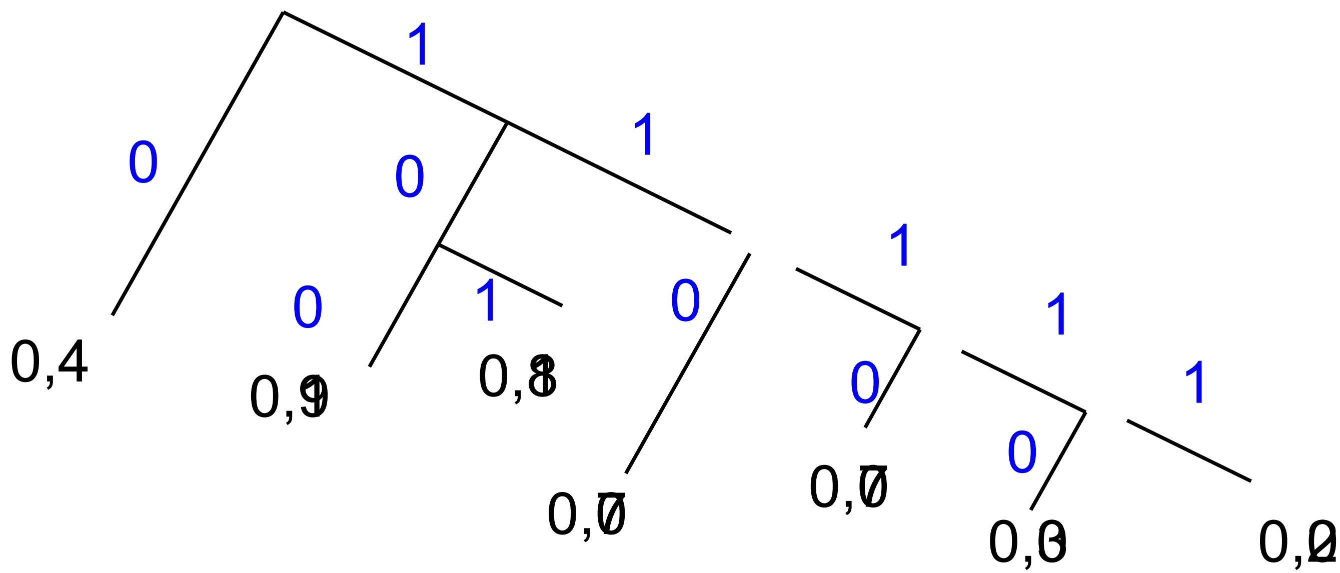
2+3 p=0,37

Puis 1 reste
seul

2.2. Codage de longueur variable: Codage de Huffman

0,44	**0,19**	**0,18**	**0,07**	**0,07**	**0,03**	**0,02**
1	**100**	**101**	**110**	**1110**	**11110**	**11111**

L(Huffman) = L(SF)= 2,23 bit/symbole



2.2. Codage de longueur variable:

Résumé

Codage Virgule, Virgule « G », Shannon-Fano, Huffman

De façon théorique, le codage d'Huffman est le plus « optimal » en termes de longueur moyenne.

2.2. Codage arithmétique:

principe une séquence de S symboles est traduite par un intervalle entre 0 et 1 de longueur égale à la proba de S sur I = ceil(-log2(p(S)) bits

$$x \in [0,1] \rightarrow x = \sum_{i=1}^{\infty} d_i 2^{-i} ; avec\, d_i \in \{0,1\}$$

0.25 ^ 0.01 **0.43 ^ 0.0110111000...**

0.125 ^ 0.001 **0.71 ^ 0.101101111...**

0.625 ^ 0.101

Attention il peut exister plusieurs developpements 2-adique (eg 0.25 ^ 0.0011111....) on prend le plus court

2.2. Codage arithmétique

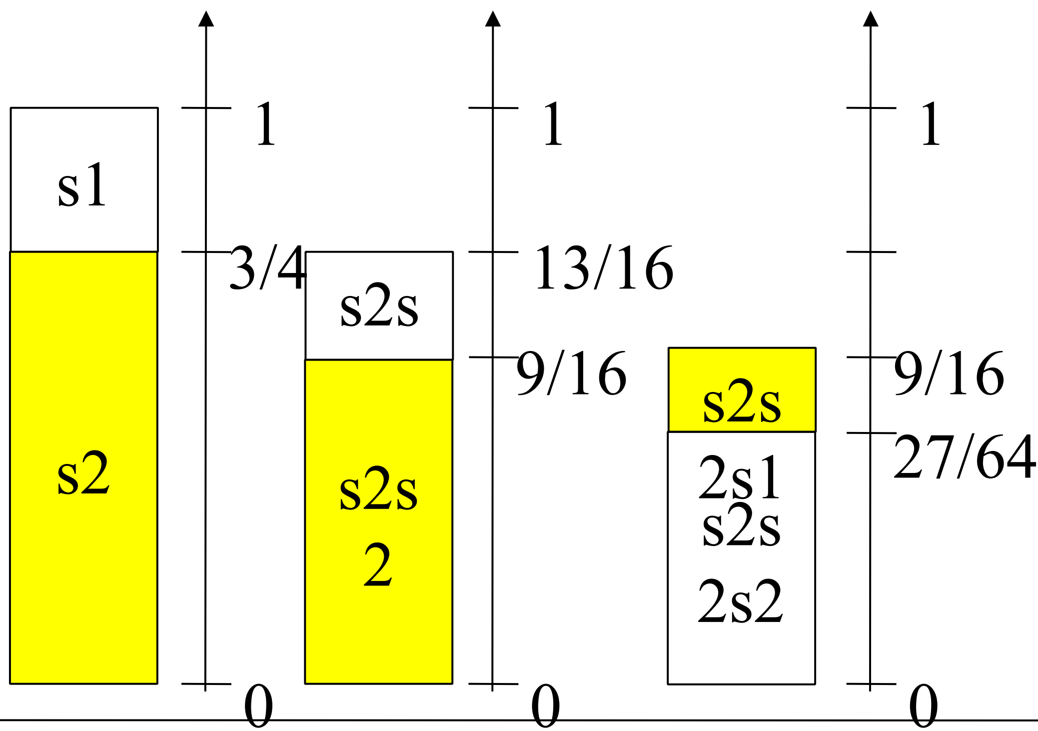
exemple : 2 symboles : s1 p1=1/4 et s2 p2=3/4

On envoie s2s2s1 ^ intervalle entre 9/16 (=1/2 +1/16 ^ 1001)

Et 27/64 (=1/4+1/8+1/16+1/64 ^ 011101)

Sur 3 bits : 4/8 ^ 100

$$c_m = \sum_{i=1}^{m-1} p_i$$



2.2. Codage arithmétique

$$probas\ cumulées\ c_m$$

Algo IC (intervalle de codage)

$$c_1 = 0 \quad c_{(M+1)} = 1$$

Calcul probas cumulées

$$c_m = \sum_{i=1}^{m-1} p_i$$

Bic(0)= 0 (base) , Lic(0)=1 (longueur)

Tant que pafini

- **Si xn = am (de proba pm) alors**

  Bic(n)] Bic(n-1)+cm.Lic(n-1)

  Lic(n)] pm. Lic(n-1)

rem : IC(n) s'emboite dans IC(n-1)

Lic(n) = p1.p2 ...pm

IC(n) codé par b12^-1+ ...+ bNB avec NB bits

2.2. Décodage arithmétique

exemple : 2 symboles : s1 p1=1/4 et s2 p2=3/4

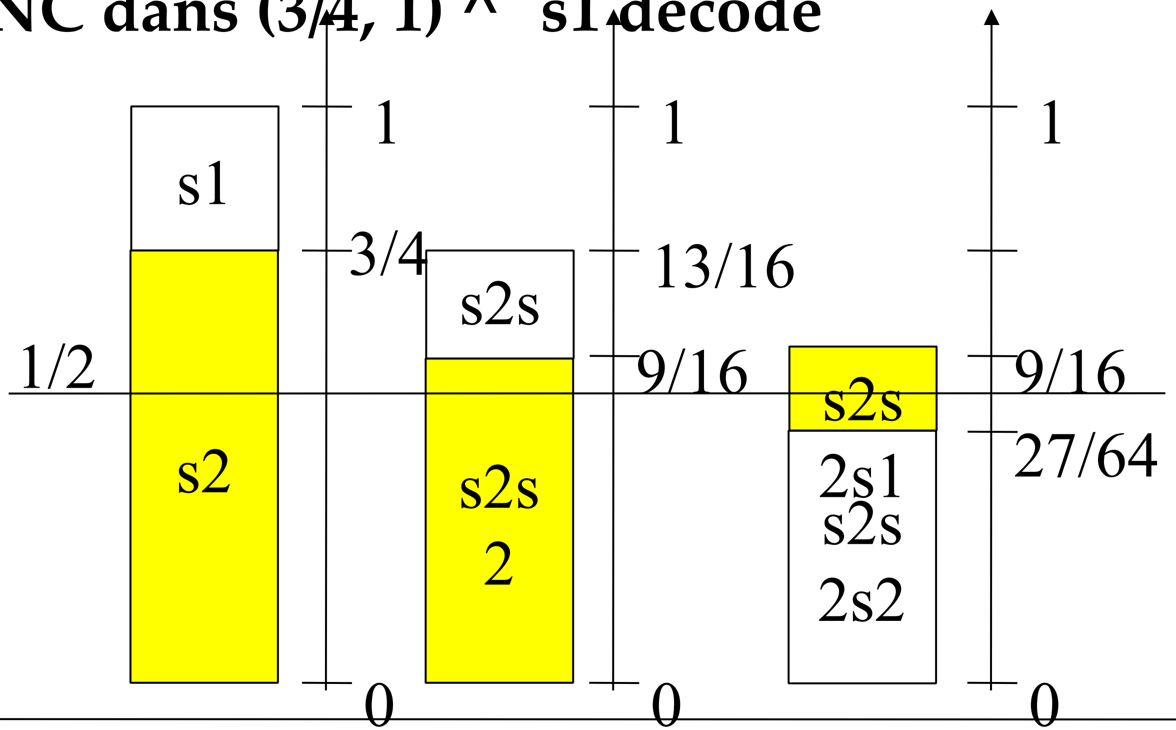
Code 100 NC=1/2 (Long=3)

NC dans (0,3/4) ^ s2 décodé NC] NC/3/4 = 2/3

NC dans (0,3/4) ^ s2 décodé NC] NC/3/4 = 8/9

NC dans (3/4, 1) ^ s1 décodé

$$c_m = \sum_{i=1}^{m-1} p_i$$



2.2. Codage arithmétique

$$probas\ cumulées\ c_m$$

Algo décodage

$$c_1 = 0 \; c_{(M+1)} = 1$$

Le code reçu est traduit en fraction NC

1)Trouver m tq NC dans intervalle (cm, cm+1) $c_m = \sum_{i=1}^{m-1} p_i$

=> donne le symbole am

2) mise à jour NC] (NC - cm-1)/pm

intervalle (cm, cm+1) dilaté sur (0,1)

rem : la dilatation permet de supprimer l'influence de am

Le décodeur doit savoir (en plus des pm) le nb de symboles à décoder

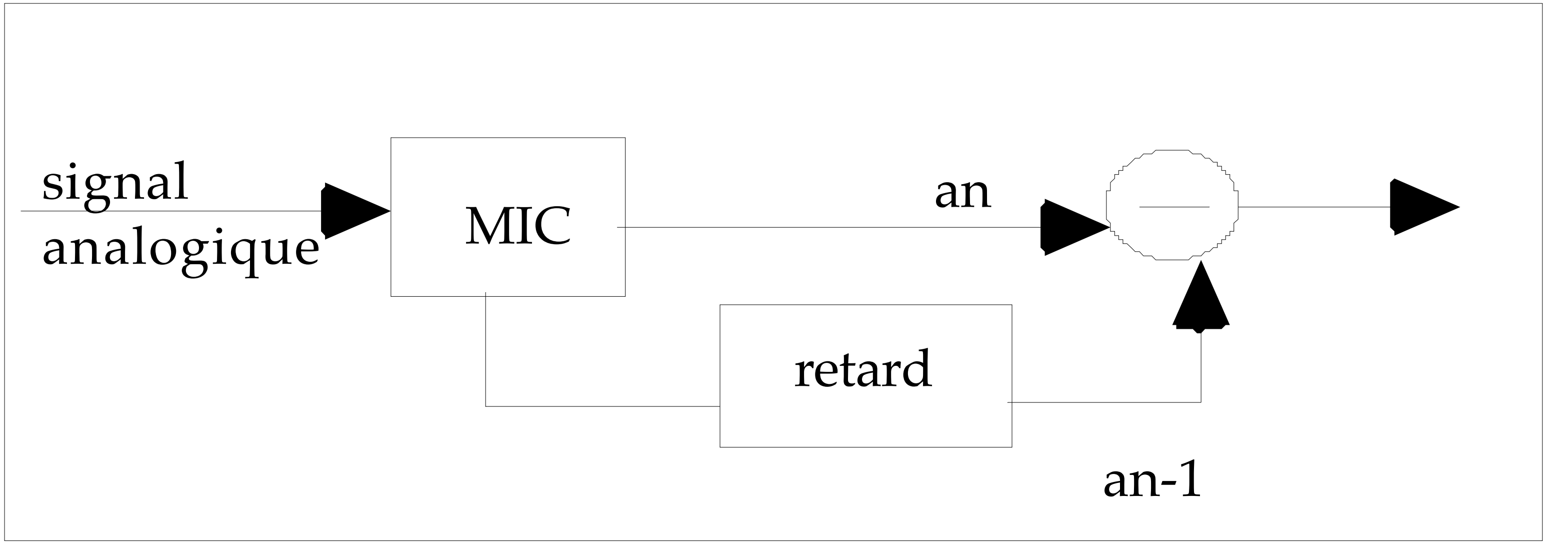
2.3. Codage MICD

Notion de Codage MIC (modulation impulsion codées) et MICD (différentiel)

Ce codage permet de passer simplement d'un signal analogique à un signal numérique en discrétisant chaque échantillon (son, image) sur un nombre fixe de bits. L'astuce est d'exploiter la corrélation de deux échantillons successifs en effectuant une différence : c'est ainsi que l'on décorrèle le signal (pas completement toutefois).

2.3. Codage MICD

Notion de Codage MIC (modulation impulsion codées) et MICD (différentiel)



2.3. Codage MICD

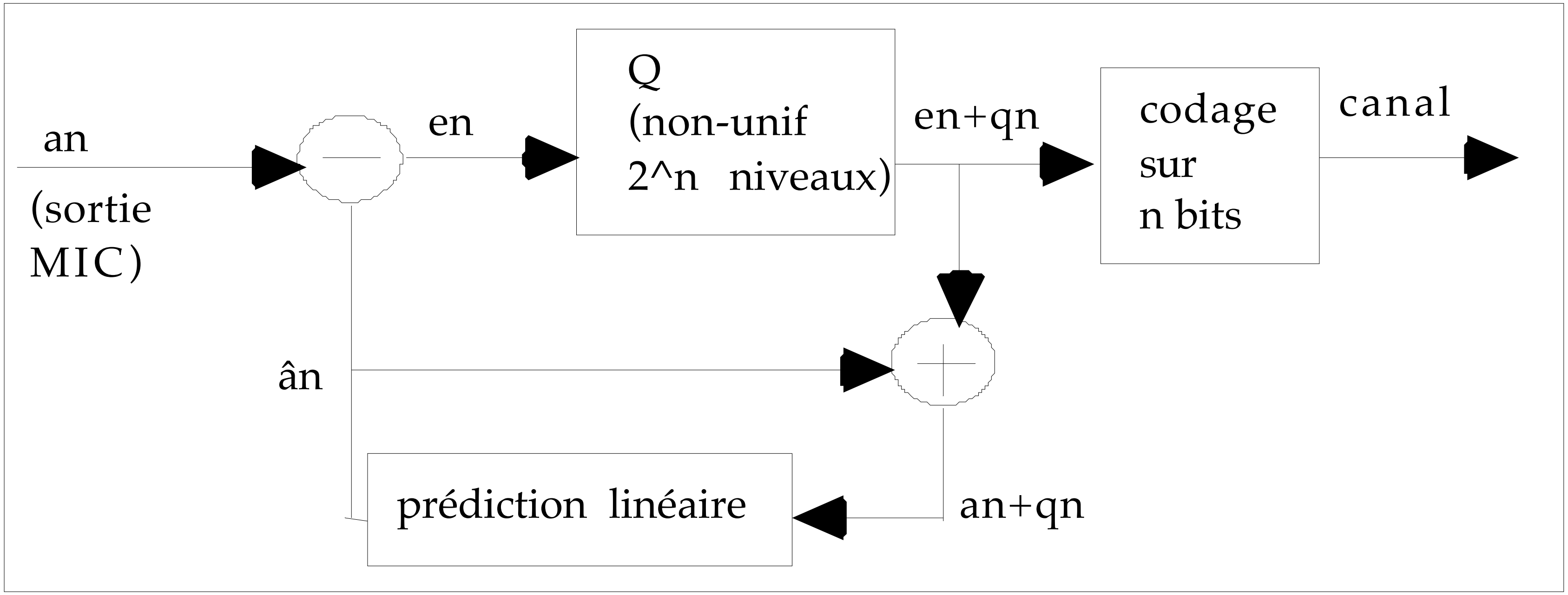
Notion de Codage MIC (modulation impulsion codées) et MICD (différentiel)

On obtient ainsi un signal en sortie dont l'entropie est inférieur à l'entropie du signal à la sortie du MIC lorsque l'on a Prob{an = an-1 | an-1 } =1 - ε. On a ensuite deux possibilités : soit en code en longueur variable le résultat (cf paragraphe suivant) soit on quantifie et on code le signal différentiel : c'est la MICD. Attention ici : la phase de quantification va faire perdre de l'information de manière irréversible.

2.3. Codage MICD

Codeur MICD

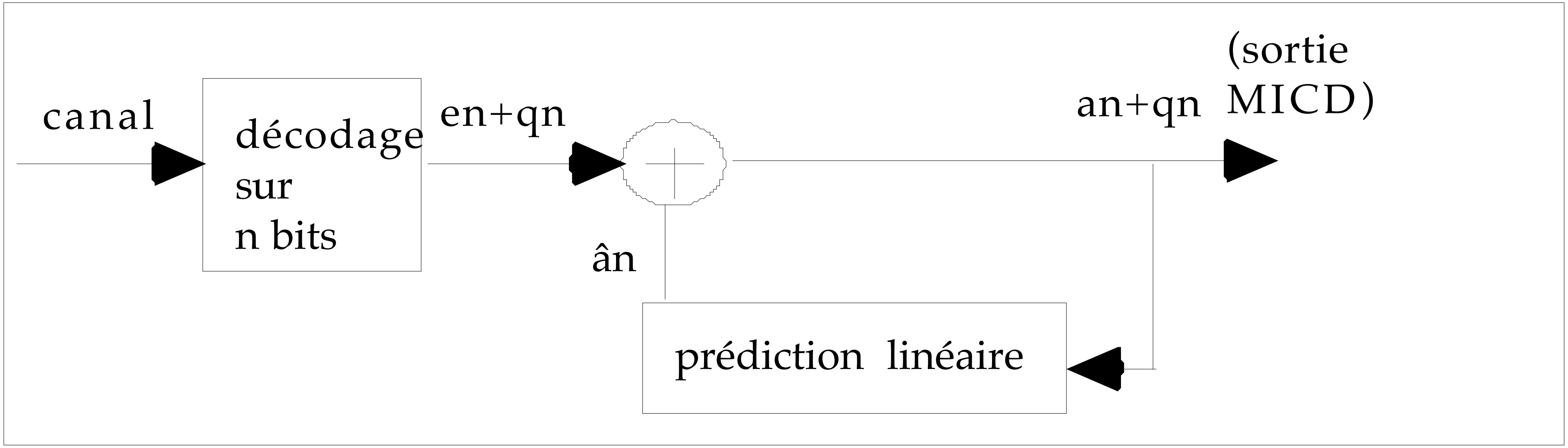
en = an —ân, qn : bruit de quantification



2.3. Codage MICD

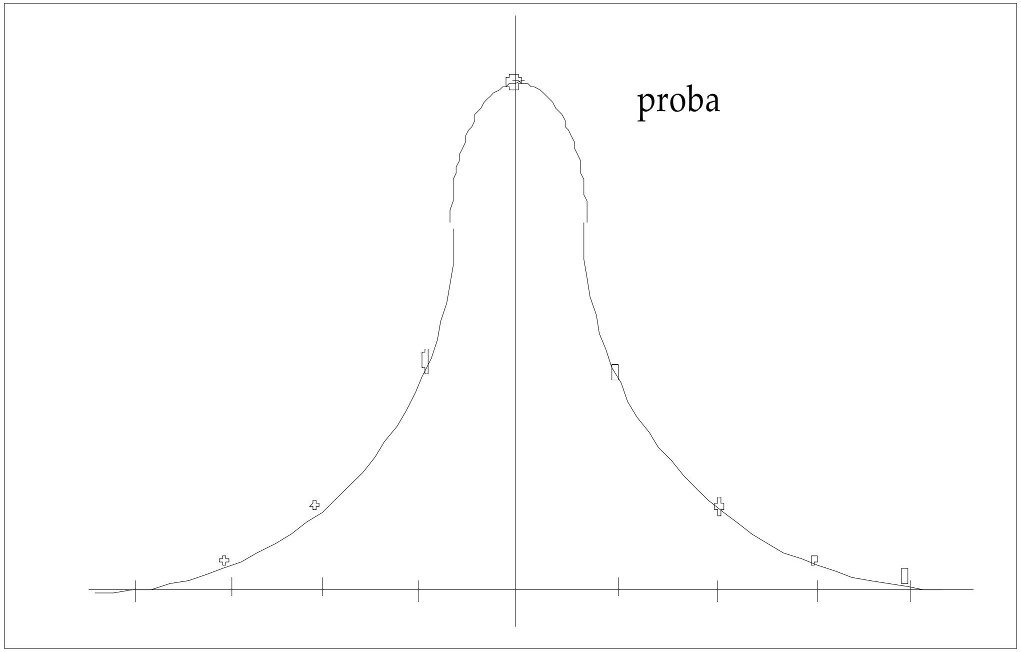
Décodeur MICD

en = an —ân, qn : bruit de quantification

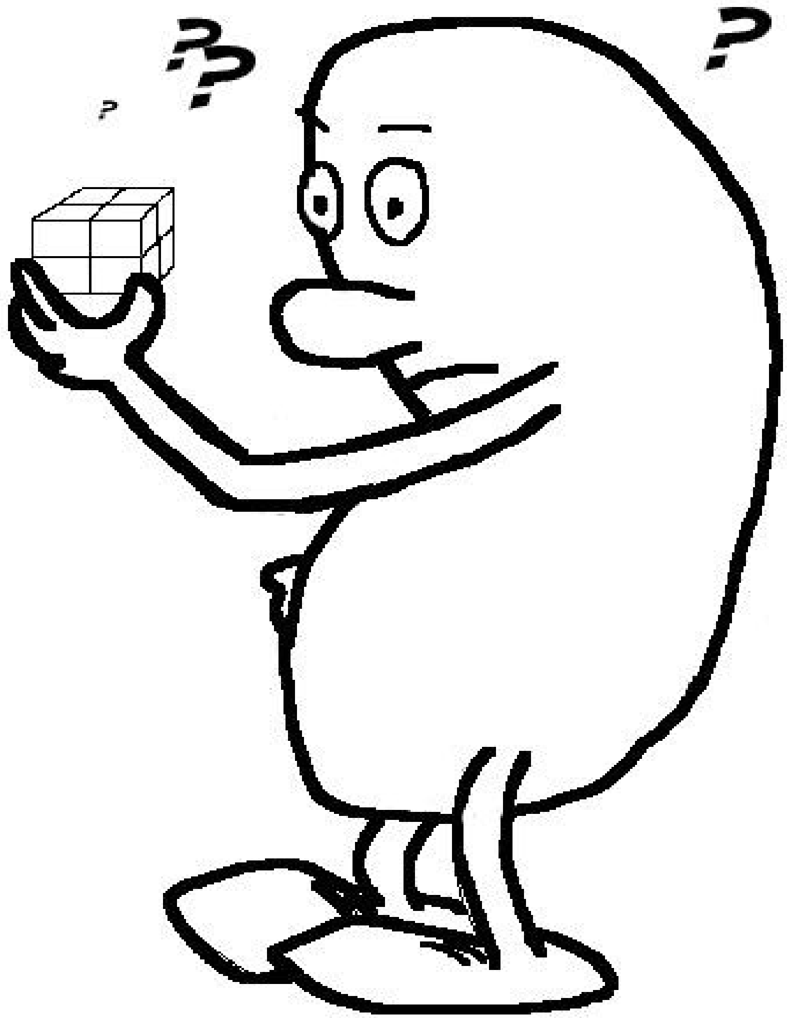


2.3. Codage MICD

Décodeur MICD : Proba de sortie d'un MICD avec quantification sur 3 bits



Théorie de l' Information



- **merci**