1. Définitions :

Canal : dispositif permettant d'acheminer un message entre deux points distants.

Objectif : trouver des algorithmes de codage simples et performants, qui permettent de détecter et de corriger le maximum d'erreurs, tout en allongeant le moins possible les mots. La protection contre les erreurs (dues à la transmission sur le canal) est assurée en ajoutant de la redondance.

Capacité du canal (C) : Vitesse maximale (ou le taux le plus élevé) de la transmission

$$C = max_{p(x)} I(X,Y) = \max[\mathrm{H(X)} - \mathrm{H(X/Y)}]$$

Tel que *I(X,Y)* est l'information mutuelle donnée par cette formule:

$$I(X,Y) = \sum_{x \in X,\ y \in Y} p(x,y) log_2 \frac{p(x,y)}{p(x)p(y)}$$

Exemple 1 : Canal binaire sans bruit $C=1$ bit

Exemple 2 : Canal binaire bruité $C=1-H(a)$ bit

Vitesse de transmission (débit moyen) de l'information : Soit une source émettant *rs* symboles/seconde, La vitesse de transmission de son information vaut :

Vitesse de Transmission = *rs* * H(X) bits/sec

2. Deuxième théorème de Shannon :

On peut transmettre de l'information de façon fiable en utilisant un code correcteur d'erreur de taux de transmission inferieur à la capacité du canal utilisé.

Le 2ème théorème de Shannon énonce une condition d'adéquation entre la source et le canal pour obtenir un taux d'erreur aussi faible que souhaité.

3. Classifications des codes canal :

Il existe différentes classes de codes canal :

A. En bloc :

- Détection d'erreur : parité, codes polynomiaux.
- Détection/Correction d'erreur : codages de Hamming, codes linéaires, codes cycliques, Cyclic Redundancy Check (CRC), codes BCH (Bose Chandhuri Hocquengheim).

B. Convolutionnels (ou récurrents)

C. Turbo-codes

4. Codage détection d'erreurs :

Un codeur est dorénavant décrit par : code(n,k)

n= nombre de bits à transmettre

k= nombre de bits de mot original.

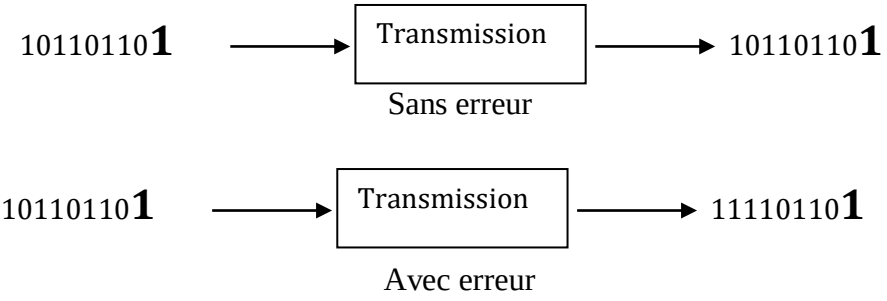
d= distance minimale = n-k.

4.1. Parité : cette technique peut détecter une seule erreur sans correction, en ajoutant un bit de parité à la fin comme suit :

- Ajouter '0' si la somme des '1' est paire
- Ajouter '1' si la somme des '1' est impaire

Exemple :

Soit le mot suivant '10110110' ↦ parité impaire donc an ajoute '1'



Mais on ne peut pas ni localiser ni corriger cette erreur avec cette technique.

5. Codage détecteur/correcteur d'erreurs :

5.1. Distance de Hamming :

La distance de Hamming entre deux mots est le nombre *d(x,y)* de symboles qui diffèrent.

Exemple :

- la distance entre '10100111' et '10111111' vaut 2,
- tandis que celle entre '10100111' et '11000001' vaut 4.

Remarque :

- Plus la distance minimale est élevée, plus le code peut corriger d'erreurs.
- D'une manière générale, $d_{min} = n-k$.
- Nb max des erreurs détectables : $e_D = d_{min} - 1$
- Nb max des erreurs à corriger : $e_c = \lfloor (dmin - 1)/2 \rfloor$ ($\lfloor . \rfloor$ = partie entière)

5.2. Poids d'un code

Le poids (W_i) d'un mot codé est par définition le nombre de caractères non nuls que contient ce mot.

Exemple 01 : 11001000 est de poids W=3

Exemple 02 : 10011010 est de poids W=4

Exemple 03 : 00000000 est de poids nul.

5.3. Code de Hamming :

Les codes de Hamming sont des codes binaires auto-correcteurs qui ont la propriété :

$$(n, k, d_{min}) = (2^m - 1, 2^m - 1 - m, m)$$

Il consiste à ajouter les bits de parités (bits de contrôle) en puissance de 2 de droite à gauche (2^0, 2^1, 2^2, 2^3, 2^4, …...).

Exemple : code (7, 4, 3) = d7 d6 d5 p4 d3 p2 p1

Ici on a m=3 bits de parité pour contrôler k=4 bits de mots.

Comment remplir les bits de parité p1, p2 et p3 (o : pair, 1 : impair):

Etape 1 : vérifier un, sauter un ↦ trouver la parité des bits : 1, 3, 5, 7

Etape 2 : à partir de p2 vérifier deux, sauter deux ↦ trouver la parité des bits : 2, 3, 6, 7

Etape 3 : à partir de p4 vérifier quatre, sauter quatre ↦ trouver la parité des bits : 4, 5, 6, 7

Exercice 01 : Coder cette information '0001' selon Hamming.

Sol 01 :

D7	D6	D5	P4	D3	P2	P1
0	**0**	**0**	**?**	**1**	**?**	**?**
7	6	5	4	3	2	1

P_1= ? ↦ On vérifie la parité de : P_1 D_3 D_5 D_7 = 0 0 1 P_1 ⇒ $P_1 = 1$

P_2= ? ↦ On vérifie la parité de : P_2 D_3 D_6 D_7 = 0 0 1 P_2 ⇒ $P_2 = 1$

P_4= ? ↦ On vérifie la parité de : P_4 D_5 D_6 D_7 = 0 0 0 P_4 ⇒ $P_4 = 0$

Donc le code Hamming de l'information '0001' est '0000111'

Exercice 02 : vérifier cette information reçue '1011100'.

Sol 02 :

D7	D6	D5	P4	D3	P2	P1
1	**0**	**1**	**1**	**1**	**0**	**0**
7	6	5	4	3	2	1

Etape 1 : vérifier 1, 3, 5, 7 :

P1 + D3 + D5 + D7 = 0+1+1+1=3 Donc c'est faux, A1=1.

Etape 2 : vérifier 2, 3, 6, 7 :

P2 + D3 + D6 + D7 = 0+1+0+1=2 Donc c'est juste, A2=0.

Etape 3 : vérifier 4, 5, 6, 7 :

P4 + D5 + D6 + D7 = 1+1+0+1=3 Donc c'est faux, A3=1.

Etape 4 : Lire $A_3A_2A_1$ en décimal pour localiser l'erreur et la corriger :

$A_3A_2A_1 = (101)_2 = 5$ (le bit erroné se trouve dans la position 5)

Donc, l'information correcte est '1001100'.

Exercice 03 : Trouver le code de Hamming de ce message '01011001'

6. Codes linéaires :

Un codage est dit linéaire (**LBC** : Linear Block Code)) quand le code *C* vérifie :

$$C(u1 \oplus u2 \oplus \ldots \oplus up) = C(u1) \oplus C(u2) \oplus \ldots \oplus C(up)$$

Où u_1, u_2,…, u_p sont les messages de source.

X	Y	S = X ⊕ Y
0	0	0
0	1	1
1	0	1
1	1	0

Table de vérité de XOR

Exemple 01 : codage de parité C(3,2)

Message u_i	***séquence***	***Code*** C_i
$u1$	00	000
$u2$	01	011
$u3$	10	101
$u4$	11	110

$C(u1 \oplus u2) = C(00 \oplus 01) = C(01) = 011$ ----------(1)

$C(u1) \oplus C(u2) = C(00) \oplus C(01) = 000 \oplus 011 = 011$ --------(2)

Les deux parties (1) et (2) sont égales, donc le codage parité est linéaire.

Exemple 02 : codage de Hamming C(7,4,3)

Message u_i	***séquence***	***Code C_i***
*u*1	1000	1001011
*u*2	0100	0101010
*u*3	0001	0000111
*u*4	1101	1100110

C(u1⊕u2⊕u3) = C(1000⊕0100⊕0001) = C(1101) = 1100110 ----------(1)

C(u1) ⊕C(u2) ⊕C(u3) = C(1000) ⊕ C(0100) ⊕ C(0001)

= 1001011 ⊕ 0101010 ⊕ 0000111 = 1100110 ----------(2)

Les deux parties (1) et (2) sont égales, donc le codage parité est linéaire.

7. Codes Systématiques :

Un code **LBC** est dit systématique lorsque les bits de contrôle $c_1, c_2, \ldots, c_m$ ($m = n - k$) s'ajoutent directement aux k bits de message. Le code résultant est de longueur ($n = k + r$) comme suit $[d_{k,\ldots}, d_2, d_1, c_m, \ldots, c_2, c_1]$.

Au récepteur, le contrôle est alors simple : on calcule les bits de contrôle (clé) correspondant au mot formé par les k premiers bits du message. Si cette clé est différente de celle qui se trouve en fin du mot reçu, il y une erreur.

Exemple : codage de Hamming C(7,4,3)

d7 d6 d5 p4 d3 p2 p1 ---------- ce code n'est pas systématique

d4 d3 d2 d1 p3 p2 p1 ---------- ce code est systématique

Message u_i	***séquence***	***Hamming non systématique***	***Hamming systématique***
*u*1	1000	1001011	1000 111
*u*2	0100	0101010	0100 110
*u*3	0001	0000111	0001 011
*u*4	1101	1100110	1101 010

8. Matrice génératrice d'un code :

La matrice génératrice d'un code linéaire systématique a la forme suivante :

$$G = (I_k \mid P), \quad \text{où } [I_k] = k \times k, \quad [P] = k \times d;\ d = n-k$$

Où *Ik* est une matrice identité de taille k et *P* est une matrice de parité.

Pour coder un message (*m*) de taille *k* (c.-à-d. : [m_1 m_2 m_3….m_k]), il suffit de le multiplier par la matrice génératrice *G*. Le résultat est le code de taille *n* :

$$\text{Code} = m \,.G$$

[*m*]=1×*k*, [*G*]=*k*×*n*, [*code*]=1×*n*. Rappelons que l'addition se fait en modulo 2.

Exemple 01 : codage de parité C(4,3)

Message u_i	***séquence***	***Code*** C_i
*u*1	000	0000
*u*2	001	0011
*u*3	010	0101
*u*4	100	1001

$$G = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{array}\right]$$

Donner le code de 011 en utilisant le même codage de parité C(4,3) :

$$C(011) = [011].G = [011].\left[\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{array}\right] = [0110]$$

Exemple 02 : codage de Hamming systématique C(7,4,3)

Message u_i	***séquence***	***Code*** C_i
*u*1	1000	1000 111
*u*2	0100	0100 110
*u*3	0010	0010 101
*u*4	0001	0001 011

$$G = \begin{bmatrix} 1000 & 111 \\ 0100 & 110 \\ 0010 & 101 \\ 0001 & 011 \end{bmatrix} \text{ où } P = \begin{bmatrix} 111 \\ 110 \\ 101 \\ 011 \end{bmatrix} \text{ et } I_k = \begin{bmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \end{bmatrix}$$

Donner le code de (1111) en utilisant le même codage de Hamming C(7,4,3) :

$$C(1111) = [1111].G = [1111].\begin{bmatrix}1000 & 111\\ 0100 & 110\\ 0010 & 101\\ 0001 & 011\end{bmatrix} = [1111\ 111]$$

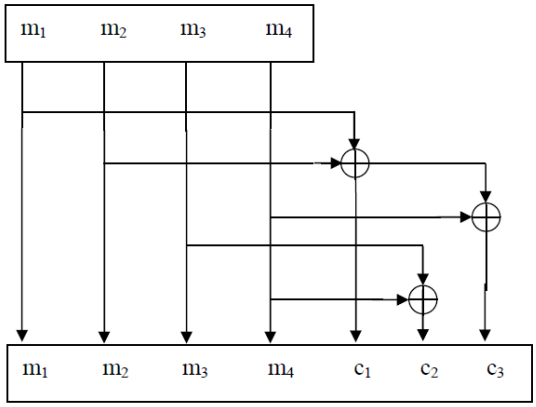
Exercice :

Soit le code *C* (7,4) qui au vecteur d'information $m = (m_1,m_2,m_3,m_4)$ associe le mot de code $(m_1,m_2,m_3,m_4,c_1,c_2,c_3)$, avec $c_1 = m_1+m_2$, $c_2 = m_3+m_4$, et $c_3 = m_1+m_2+m_4$.

1- Tracer le circuit du codeur.
2- Montrer si ce code est linéaire.
3- Trouver la matrice génératrice *G*, donner le code correspondant au message (1 0 1 0).
4- Maintenant, on considère ce codage non-systématique selon cet ordre $(m_3,c_1,,m_1,c_3,m_2,m_4,c_2)$, déduire la nouvelle matrice de génération G_2.
5- Que devient le code correspondant au message (1 0 1 0).

Solution :

1- Le circuit du codeur :



2- Confirmant que ce code satisfait la condition de linéarité. Prenons deux messages de source $m= (m1,m2,,m3,m4)$ et $u= (u1,u2,,u3,u4)$. On a :

$C(m+u) = (\ m1+ u1,m2+ u2,,m3+ u3,m4+ u4,\ m1+ u1+ m2+ u2,m3+ u3+ m4+ u4,\ m1+ u1+m2+u2 +m4+ u4)$

$= (m1,m2,,m3,m4,\ m1+m2,m3+m4,\ m1+m2+m4)+ (u1,\ u2,u3,u4,u1+u2,\ u3+u4,u1+u2+u4)$

$= C(m)+ C(u)$

3- La matrice génératrice de ce code est donnée par :

$$G = \begin{bmatrix}1000 & 101\\ 0100 & 101\\ 0010 & 010\\ 0001 & 011\end{bmatrix}$$

Le code correspondant au message (1 0 1 0) est obtenu comme suit :

$$code = message.G = [1010].\begin{bmatrix}1000 & 101\\0100 & 101\\0010 & 010\\0001 & 011\end{bmatrix} = [1\ 0\ 1\ 0\ 1\ 1\ 1]$$

4- Un codage non-systématique est donné par (*m3,c1,m1,c3,m2,m4,c2*) ↦ La matrice génératrice est obtenue par simple permutation des colonnes de la matrice *G* calculée en haut selon les positions de bits de message et de bits de contrôle. On obtient ainsi la matrice *G*2 suivante :

$$G2 = \begin{bmatrix}0111 & 000\\0101 & 100\\1000 & 001\\0001 & 011\end{bmatrix}$$

Le nouveau code du message (1 0 1 0) selon *G2* est :

$$code2 = message.G2 = [1010].\begin{bmatrix}0111 & 000\\0101 & 100\\1000 & 001\\0001 & 011\end{bmatrix} = [1\ 1\ 1\ 1\ 0\ 0\ 1]$$

9. Matrice de Contrôle et la notion Syndrome :

- On a **G** = [I_k, P] la matrice génératrice. Tout code en bloc admet une matrice de contrôle **H** telle que :

$$G.H^T = 0$$

- On définit la matrice de contrôle **H** par :

$$H = [P^T, I_{n-k}]$$

- La matrice **H** sert à vérifier si un mot code **C** a été généré par **G**. En effet, il a été démontré qu'un message n'est un mot du code en blocs linéaires C(n,k), que si et seulement si :

$$C.\ H^T = 0$$

- Soit **C** transmis sur un canal bruité et soit **MR** le mot correspondant reçu. On a alors:

$$MR = C + Er, \quad Er \text{ étant le vecteur erreur.}$$

- Le syndrome du mot reçu **MR** est le vecteur **S** défini par :

$$S(MR) = MR \times H^T$$

- Le syndrome est nul si et seulement si **MR** ∈ **C**.

Exemple : code (7,4)

Soit un mot (1101) et la matrice génératrice G suivante :

$$G = \begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$$

Le mot de code (1101) = (1101)*G = (1101) * $\begin{bmatrix} 1000 & 101 \\ 0100 & 111 \\ 0010 & 110 \\ 0001 & 011 \end{bmatrix}$ = 1101001

La matrice de contrôle H = [P^T, I_3] = $\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$ ⇒ H^T = $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

S(1101001) = (1101001) * $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ = 000 ⇒ Code(1101001) ∈ C(7,4)

Cas d'une erreur :

- Si le code reçu est MR = Er + C alors :

$$S(MR) = (Er + C)*H^T = Er*H^T + C*H^T = Er*H^T \quad (\text{puisque } C*H^T = 0)$$

- Donc le syndrome ne dépend que de l'erreur.
- Chaque ligne de H^T est le syndrome d'une erreur simple. On construit un tableau standard pour corriger les erreurs.

Exemple précédent :

Code du mot (1101) = 1101001, mais message reçu **MR** = 1101101

S(1101101) = (1101101) * $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ = 100

Erreur probable	Syndrome	Bit erroné
0000000	000	Pas d'erreur
1000000	101	Bit 1
0100000	111	Bit 2
0010000	110	Bit 3
0001000	011	Bit 4
0000100	100	Bit 5
0000010	010	Bit 6
0000001	001	Bit 7

Syndrome(1101101) = 100, à partir de ce tableau Er = 0000100

Maintenant, on fait la correction :

C = MR $\oplus$ Er = 1101101 $\oplus$ 0000100

Code corrigé C = **1101001**

10. Code cyclique de redondance (CRC) (détecteur d'erreur) :

Le code cyclique de redondance (**CRC** : Cyclic Redundancy Check) est un code linéaire systématique détecteur d'erreurs. La méthode de calcul de bits de redondance constituant le CRC est décrite ci-après.

Soit la suite binaire $M=m_k....m_2m_1$ constituant le message de source à transmettre sur le canal.

10.1. Emission d'un CRC :

1. Message binaire à émettre M = 1111011101, (ici $\boldsymbol{k}$ = **10**)
2. Choisir un polynôme générateur d'ordre $\boldsymbol{m}$, $G(x) = x^4+x^2+x^1$, (ici $\boldsymbol{m}$ = **4**)
3. Transformer de $G(x)$ en un mot binaire comme suit : $G(x) = \mathbf{1}.x^4+\mathbf{0}.x^3+\mathbf{1}.x^2+\mathbf{1}.x^1+\mathbf{0}.x^0$,

   Le mot binaire = **1 0 1 1 0**
4. Ajouter de $\boldsymbol{m}$ zéros au message binaire à transmettre M_e ($\boldsymbol{k+m}$ *bits*) = 1111011101**0000**,
5. Faire la division binaire (**XOR** binaire) :

```
1 1 1 1 0 1 1 1 0 1 0 0 0 0
1 0 1 1 0
---------------------------
0 1 0 0 0 1 1 1 0 1 0 0 0 0
  1 0 1 1 0
  ---------------------------
  0 0 1 1 1 1 1 0 1 0 0 0 0
    1 0 1 1 0
    ---------------------------
    0 1 0 0 1 0 1 0 0 0 0
      1 0 1 1 0
      ---------------------------
      0 0 1 0 0 1 0 0 0 0
        1 0 1 1 0
        ---------------------------
        0 0 1 0 0 0 0 0
          1 0 1 1 0
          ---------------------------
          0 0 1 1 0 0
```

6. Continuer la division jusqu'à ce que le mot obtenu soit inférieur au polynôme générateur.
7. Le reste de division (en **vert**) représente le CRC à ajouter au mot avant transmission, donc le mot de code à transmettre sur le canal est :

   1111011101**1100**

10.2. Détection d'erreur à la réception d'un CRC :

1. Supposons que le mot reçu est : 11110001010101,
2. La division binaire de ce mot avec le polynôme générateur donne le résultat suivant :

```
1 1 1 1 0 0 0 1 0 1 0 1 0 1
1 0 1 1 0
-----------------------
0 1 0 0 0 0 0 1 0 1 0 1 0 1
  1 0 1 1 0
  -----------------------
  0 0 1 1 0 0 1 0 1 0 1 0 1
      1 0 1 1 0
      -----------------------
      0 1 1 1 1 0 1 0 1 0 1
        1 0 1 1 0
        -----------------------
        0 1 0 0 0 1 0 1 0 1
          1 0 1 1 0
          -----------------------
          0 0 1 1 1 0 1 0 1
              1 0 1 1 0
              -----------------------
              0 1 0 1 1 0 1
                1 0 1 1 0
                -----------------------
                0 0 0 0 0 1
```

Le reste de la division n'est pas **nul (zéro)** ; il existe alors des erreurs dans le mot reçu.

Exercice : On a le message *M* suivant : *M*=1101101, le polynôme générateur G(x) est donné par $G(x)= x^4 + x^2 + 1$.

1- Vérifier si le mot reçu suivant 1101101 1000 est arrivé sans erreur.
2- Calculer le CRC correspondant. (solution : **1101101 1011**)

11. Turbo-codes :