Chapitre 2 : Codage et protection contre les erreurs (Couche 2 du modèle OSI)

2.1 Codage et représentation de l'information textuelle :

Un code est caractérisé par sa longueur, qui n'est autre que le nombre de bits représentant chaque caractère. Les codes les plus fréquents en téléinformatique ont des longueurs de 5, 6 ou 7 ou 8 bits permettant de coder respectivement 32, 64, 128 ou 256 caractères différents.

- Code **CCITT** (comité consultatif international télégraphique et téléphonique-) Ce code, encore appelé code Baudot est utilisé pour la transmission sur le réseau télégraphique (TELEX). C'est un code à 5 bits qui ne permet donc que 32 combinaisons
- Code **DCB** (décimal code binaire) ce code à 6 bits a été très utilisé pour la transmission avec des terminaux asynchrones.
- Code **CCITT n°5** à 7 bits, appelé aussi code **ASCII** (American Standard Code for Information Interchange) ou encore alphabet international n°5 est très utilisé et permet de coder 128 caractères
- Code **EBCDIC** ce code (Extended Binary Coded Decimal Interchange Code) à 8 bits et très utilisé du fait que chaque caractère est représenté par un octet.

2.2 Problématique :

La transmission de données entre machines distantes s'effectue en général en série par bit et la communication entre équipements informatiques donne lieu à l'échange de messages qui comportent un nombre variables de bits. Les messages qui constituent la partie utile de l'information sont complétés par des informations de contrôle et de supervision telles que l'adresse du destinataire et une séquence de contrôle d'erreurs.
Entete (header) + texte (text) + terminaison (trailer).

Les ordinateurs, comme les êtres humains, sont enclins à l'erreur occasionnellement. Lors des échanges de données, les canaux de transmission imparfaits entraînent souvent des erreurs. Les erreurs de transmission sont due aux perturbations électromagnétiques (le démarrage de moteurs, le passage prés d'un circuit à courant fort, les orages, …etc.), au câblage mal isolé et aux effets de distorsion. Le taux d'erreur sur un canal de transmission est égale au (nombre de bits erronés / le nombre de bits émis). Il est de l'ordre de 10^{-9} pour les réseaux locaux filaires ; ce taux est nettement plus élevé pour les réseaux sans fil. Il existe deux stratégies possibles pour détecter les erreurs, la destination peut :

- Détecter les erreurs, puis demander une retransmission (c'est les codes détecteurs d'erreurs)
- Détecter et corriger les erreurs (les codes correcteurs d'erreurs)

Pour détecter ces erreurs, on utilise des méthodes de détection et éventuellement de correction des erreurs.

Principe général :

- Chaque suite de bits à transmettre est augmentée par une autre suite de bits dite de **redondance** ou de **contrôle.**
- Pour chaque suite de k bits transmise, on ajoute r bits. On dit alors que l'on utilise un code C(n,k) avec n = k +r .
  n: taille de l'alphabet du code ;
  k : taille de l'information utile.
- À la réception, les bits ajoutés permettent d'effectuer des contrôles à l'arrivée. Dans le cas d'une réception sans anomalie, il suffira d'extraire l'information utile.

2.3 Généralités sur les codes :

Un code de longueur N est un ensemble C de séquences de N bits. Une séquence de N bits n'appartenant pas à C sera dite invalide.
A la réception d'une séquence de N bits, deux cas sont possibles :

• La séquence correspond à un élément du code et la transmission sera considérée comme étant correcte (on n'est pas sûr de l'exactitude de la transmission),

• La séquence n'est pas valide. On est en présence d'une erreur (c'est une certitude), et le récepteur peut alors soit corriger l'erreur (code détecteur et correcteur d'erreurs), soit demander une retransmission (code détecteur d'erreurs).

Opérations :

Somme modulo 2	Multiplication modulo 2
0+0=0	0.0=0
0+1=1	0.1=0
1+0=1	1.0=0
1+1=0	1.1=1

- **Définition 1** : Distance de Hamming :

La distance de Hamming notée Dh d'un code C est le minimum des distances entre 2 séquences quelconques de ce code. C'est le nombre de bits différents entre ces deux séquences c.a.d c'est le nombre de bits à 1 dans le résultat du XOR.

Ex 1: M = 10001001 et M' = 10110001 => $D_h(M, M') = 3$

10001001
10110001

00111000

La distance de Hamming d'un code permet d'évaluer son pouvoir détecteur d'erreur ainsi que son pouvoir correcteur. Un code C peut détecter des erreurs d'ordre ($D_h(C)$ -1) et peut corriger des erreurs d'ordre $[D_h(C)-1)/2]$.

- **Définition 2** : Soit X le bloc émis et X' le bloc reçu. On appelle *vecteur d'erreur* $E = X \oplus X'$. Si $E = 0$, pas d'erreur. Si $E \neq 0$, il y a une erreur, localisé sur les bits 1 de E.

2.4 Techniques simples de détection d'erreurs :

● Parité transversale (ou verticale)

L'information est sectionnée en blocs de m bits qui sont généralement des caractères, et on ajoute à chaque bloc un bit de parité (r=1) de telle sorte que la somme des m+1 bits modulo 2 soit nulle (parité paire) ou égale à 1 (parité impaire).

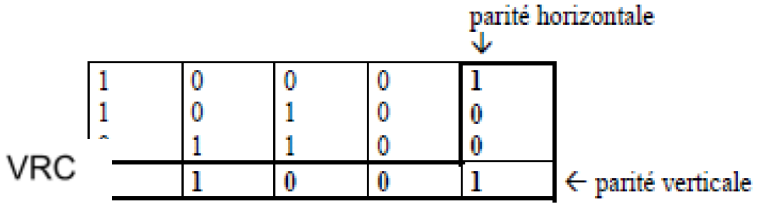
Exemple d'envoi de 4 caractères de longueur 3 (m=3).
Information utile : 110 001 011 000,
Information envoyée : 110**0** 001**1** 011**0** 000**0**.

● Parité longitudinale (ou horizontale)

On applique la même méthode aux bits de poids identique sur la totalité d'un message découpé en 'caractères'. On combine généralement la parité transversale et parité longitudinale de la façon suivante : les caractères munis de leur bit de parité transversale sont regroupés en blocs, et on ajoute à la fin de chaque bloc un caractère supplémentaire pour la parité longitudinale (ce contrôle est appelé LRC/VRC pour Vertical Redundancy Checking / Longitudinal Redundancy Checking).

Exemple d'envoi d'un bloc de 4 caractères avec un contrôle LRC/VRC:
Information utile : 110 001 011 000,
Information envoyée : 110**0** 001**1** 011**0** 000**0**.**1001**

LRC



Principe : une erreur simple modifie simultanément la parité d'une ligne et d'une colonne.
Correction : inverser le bit situé à l'intersection de la ligne et de la colonne ayan une parité incorrecte.
Exemple :

10100011
01101010
10001010
01001011

2.5 Codes linéaires systématiques

Définition : On appelle un code linéaire **C(n,k)** un code linéaire systématique un code tel que les **(n-k)** bits de contrôle sont obtenues par combinaison linéaire des **k** bits d'info utile. Formellement:

Soit $X = (x_1 \quad \cdots \quad x_n)$ le mot de code à générer

Soit $U=(u_1, .. ,u_k)$ est le vecteur d'information utile.

Soit A le vecteur de contrôle $A= (a_1,..a_{n-k})$. alors $a_i= \alpha_{1i}\, u_1+ \alpha_{2i}\, u_1+..+ \alpha_{ki}\, u_k$.

$X=(u_1,.....,u_k,a_1,..,a_{n-k})$

Ces combinaisons linéaires peuvent être formulées par un produit matricielle : $X=U.\ G$
La matrice G est une matrice de dimension (k, n), de la forme (I_k, P) avec P de dimension (k,r) avec $r=n-k$

P $\begin{pmatrix} \quad \end{pmatrix}$

1) ***Codage.***

Soit la matrice génératrice d'un code G de dimension $k \times n$ vérifie

Le calcul du mode de code X est effectuée par : $U \times G = X$.

Le sous-produit matriciel. $U\,A$ calcule les bits de contrôle.

Considérons le code $(n = 6, k = 3)$ défini par la matrice. $G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$.
Considérons un bloc de données U décrivant toutes les entrées possibles :

$U = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \\ 1 & 1 & 1 \end{pmatrix}$. On calcule $X = \left(\begin{array}{ccc|ccc} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 0 & 0 & 0 \end{array}\right)$.

2) ***Décodage.***

Soit Y le vecteur du mot de code reçu.

La vérification de l'intégrité de Y (si X=Y) se fait par le calcul du syndrome S :

Le vecteur syndrome du vecteur Y est $S(Y)=Y \times H$, tel que H est une matrice de dimension *(n, n-k)*.
H est appelée matrice de décodage et est définie par : $H=\begin{pmatrix} P \\ I_{n-k} \end{pmatrix}$

On a $S(Y)=0$, (vecteur nul) si Y appartient au code. En revanche, si on introduit une erreur, $Y = X + E$ avec E le vecteur d'erreur, alors $S(Y)= S(E)$ car $S(X)=0$.

Localisation d'erreur. Si $S(Y)=0$, alors il n'y a pas d'erreur et $Y=X$.

Si le code est correcteur $D_{min} \geq 3$

Il y a au moins une erreur, si $S(Y)\neq 0$,

On peut localiser une erreur simple en recherchant la ligne identique à $S(Y)$ dans la matrice H. La position de l'erreur dans Y correspond au rang de la ligne trouvée. On peut ensuite la corriger.

Poursuivons notre exemple avec la matrice $H^T = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ et $Y = (0\ 0\ 1\ 1\ 1\ 0)$..
On vérifie : $S(Y)=(0,0,0)$. Supposons qu'une erreur se produise sur le second bit, on reçoit $S(Y)$=(0 **1** 1 1 1 0). On calcule alors le syndrome $S(Y)=(1,0,1)$ qui correspond à la deuxième ligne de H ; l'erreur est localisée et peut être corrigé.

Circuit de codage et de décodage d'un code linéaire:

2.6 Codes polynomiaux

Ces codes, font appel à une représentation du message par un polynôme. La formule générale d'un message binaire est un polynôme de la forme :

$$\mathbf{P(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0 x^0}$$

Les coefficients a_n ne peuvent prendre que les valeurs 0 ou 1 en binaire et les x^n sont les puissances successives de 2. Si nous divisons ce polynôme par un autre polynôme binaire, le reste de la division constitue le caractère de contrôle.

Exemple: Soit à transmettre le message **1011011.** Le polynôme correspondant à la forme: $P(x)= x^6+x^4+ x^3+ x +1$
Si le polynôme diviseur est égal à : $G(x) = x^4+ x + 1$
Ce qui donne en binaire : **10011**
Il faut donc diviser **1011011** par **1001**1.

Formellement:

Supposons un code **C (n, k)** **n**: Taille du code et **k** : taille de l'info utile.

Soit **Z(x)** le polynôme représentant l'information utile à envoyer, **Z** est de degré (k-1) puisque l'information comporte k bits : $a_{k-1} x^{k-1} + ... + a_1 x + a_0 x^0$

a- Codage:

Soit **G(x)** le polynôme générateur du code C (n,k) de degré **r** tel que **r = (n-k)**
Il faut multiplier **Z(x)** par $\mathbf{x^r}$ **r** = (n-k) pour créer un décalage à gauche de **r** bits.

Exemple: soit **Z(x) 1011011** on multiplie par x_4 et on obtient : 10110110000

On divise le produit par le polynome générateur **G(x)**

On obtient: $\mathbf{Z(x)\, x^r = Q(x)\, G(x) + A(x)}$ **où:**

- **Q(x)** : polynôme quotient
- **A(x)** : reste de la division de degré: **r-1**

Donc on obtient le mot de code à envoyer comme suit: $\mathbf{Y(x) = Z(x)\, x^r + A(x) = Q(x)\, G(x)}$

Y(x) désigne un polynôme de degré (n-1).

La division est exécutée de plusieurs manières :

Méthode manuelle:

- Nous ajoutons à sa droite, autant de 0 que le diviseur comporte de bit moins un (ici 5-1 =4) ce qui donne **10110110000**.
- Puis nous exécutons la division par soustractions successives (soustractions grâce au OU-exclusif) :

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

- la dernière ligne de la division donnera le reste donc le CRC; c'est lui qui sera ajouté au message que l'on va transmettre. Y(x) **1011011 1100**

b- Décodage

A la réception, un calcul semblable s'effectue sur le mot reçu, mais il faut, ici, que le reste soit nul. Dans le
cas contraire, c'est qu'une erreur est survenue en cours de route.
Pour procéder à la vérification nous effectuons comme suit :

ÿ(x) = **y(x) + E(x)**

On Calcule le syndrome : **S(x)= ÿ(x) / G(x)** on obtient: **(G(x) A(x) +E(x)) / G(x) = 0 + E(x) / G(x)**

Aucune erreur n'est détectée si le syndrome **S(x)** =0 Sinon elle est détectée si **S(x) <> 0**

Pour qu'une erreur puisse être détectée : **E(x) <> 0** il faut que **G(x)** ne divise pas **E(x)**.

Exemple:

a) sans erreur	**b) avec erreur**
1 0 1 1 0 1 1 1 1 0 0 - 1 0 0 1 1 0 0 1 0 1 **1 1** - 1 0 0 1 1 0 0 1 0 0 **1 1** - 1 0 0 1 1 0 0 0 0 0 0 0	1 0 1 1 0 1 0 1 1 0 0 - 1 0 0 1 1 0 0 1 0 1 **1 0** - 1 0 0 1 1 0 0 1 0 1 **1 1** - 1 0 0 1 1 0 0 1 0 0 0 0 1 0 0 1 1 **0 0 0 1 1**

Si le reste est différent de 0 , le transfert s'est mal effectué.

Circuit logique d'un codeur polynomial:

2.4.3 Polynômes générateurs utilisés

Le choix du polynôme générateur est très important : de lui dépendra le pouvoir de détection de certains types d'erreur de transmission. Par exemple, il peut être intéressant d'utiliser un polynôme qui permettra de détecter des erreurs groupées. Voici quelques exemples de polynômes générateurs utilisés :

CRC-12 = X12+X11+X3+X2+X1+1,
CRC-16 = X16+X15+X2+1,
CRC-CCITT = X16+X12+X5+1.

Les deux polynômes CRC-12 et CRC-16 sont souvent utilisés et le polynôme CRC-CCITT est le polynôme normalisé (V.41) utilisé par le protocole de liaison HDLC et son efficacité se passe de commentaires puisqu'il permet de détecter :

100% des erreurs simples, doubles,
100% des paquets d'erreurs de longueur <17,

99,997% des paquets d'erreurs de longueur 17...
Et tout cela, en rajoutant seulement 16 bits de redondance sur un message dont la taille courante est de l'ordre de 1000 bits.