Codes correcteurs d'erreurs

– des fouilles sont visibles

– les nouilles sont visibles

– les nouilles sont risibles

- Généralités sur la théorie des codes
- Préambule mathématique
- Codes linéaires
- Décodage par les classes latérales

Généralités sur la théorie des codes

La théorie des codes permet

- de transmettre un message au travers d'un canal bruité comme :
  - un réseau hertzien
  - un câble téléphonique ou ethernet
  - une liaison satellite
- ou le stockages de masse (CD, ...).

A l'origine : résultat existentiel : second théorème de Shannon dit qu'il existe de bons codes

Distance de Hamming et boules

Définition Distance de Hamming

Soit B un alphabet fini; la distance de Hamming sur B est

$$d(x,y) = \begin{cases} 0 & \text{si} \quad x \neq y, \\ 1 & \text{si} \quad x = y, \end{cases} \tag{1}$$

La distance de Hamming sur B^n entre $x = x_1 x_2 \dots x_n$ et $y = y_1 y_2 \dots y_n$ est

$$d(x,y) = \sum_{i=1}^{n} d(x_i, y_i)$$

$d(x,y) = e$ ssi les mots x et y diffèrent en exactement e positions

Le poids de $x \in B^n$ est

$$\boldsymbol{w}(x) = d(x, \mathbf{0})$$

avec $\mathbf{0} = (0, 0, \ldots, 0)$

La boule de centre x et de rayon e est

$$B_e(x) = \{z : z \in B^n, d(x, z) \leq e\}$$

Lemme Soit $card(B) = q$ et $x \in B^n$, alors pour $0 \leq e \leq n$

$$card(B) = \sum_{i=1}^{e} \binom{n}{i} (q-1)^i$$

Définition Distance d entre x et y

$d(x, y) > 0$
$d(x, y) = 0$ ssi $x = y$
$d(x, y) = d(y, x)$
$d(x, y) \leq d(x, z) + d(z, y)$

Codage à longueur fixe

Coder un caractère x d'un alphabet fini A revient à le transformer en un mot y de longueur n sur l'alphabet fini B. En d'autres termes, on lui ajoute de la redondance afin de pouvoir le transmettre au travers d'un canal de communication bruité.

Ansi, une application injective

$$c : A \to B^n$$

est applée $A - B$ code, et les images des lettres de A par c sont appelées **mots du code**

$$C = \{y \in B^n : y = c(x), x \in A\}$$

$$C \subset B^n$$

On étend alors l'application c aux mots sur A par

$$c^* : A^* \to B^*$$

définie par

- $c^*(\epsilon) = \epsilon$
- $c^*(xm) = c^*(x)c^*(m)$

Afin de pouvoir corriger le mot reçu si celui-ci a eu e caractères altérés, on requiert que les mots du code soient deux à deux éloignés.

Définition Un code C de longueur n sur l'alphabet B vérifie la condition de décodage d'ordre e si $\forall x \in B^n$ il existe au plus un mot $y \in B^n$ tel que $d(x, y) \leq e$.

Cette condition est équivalente à ce que les boules fermées (pour la distance de Hamming) de rayon e et centrées sur les mots du code C soient deux à deux disjointes.

On peut alors définir à quelle condition un code peut décoder ou corrige e erreurs :

- on dit qu'un code $C \subset B^n$ **détecte** jusqu'à e erreurs si $\forall x, y \in C$, $x \neq y$, alors $d(x, y) \geq 2e$
- on dit qu'un code $C \subset B^n$ **corrige** jusqu'à e erreurs si $\forall x, y \in C$, $x \neq y$, alors $d(x, y) \geq 2e + 1$

Exemple Soit $A = \{x, y\}$ et $B = \{0, 1\}$, $n = 3$ et $e = 1$. On définit le code C comme :

$$\begin{cases} x \to 000 \\ y \to 101 \end{cases} \tag{2}$$

Les boules centrées sur x et y et de rayon 1 sont

- $B_1(x) = \{000, 100, 010, 001\}$
- $B_1(y) = \{101, 001, 111, 100\}$

- Les deux boules ne sont pas disjointes. Si on reçoit le message 100 ou 001, on ne peut pas savoir si le message émis était x ou bien y
- C est un code détecteur d'une erreur

Exemple Soit $A = \{x, y, z\}$ et $B = \{0, 1\}$, $n = 5$ et $e = 1$. On définit le code C comme :

$$\begin{cases} x \to 01110 \\ y \to 10101 \\ z \to 11011 \end{cases} \tag{3}$$

Les boules centrées sur x, y et z et de rayon 1 sont

- $B_1(x) = \{01110, 11110, 00110, 01010, 01100, 01111\}$
- $B_1(y) = \{10101, 00101, 11101, 10001, 10111, 10100\}$
- $B_1(z) = \{11011, 01011, 10011, 11111, 11001, 11010\}$

- C vérifie la condition de décodage d'ordre 1
- C est un code détecteur d'une erreur

Définition

- La **distance minimale** d'un code est la quantité

$$d(C) = min\{d(x,y) : x, y \in C, x \neq y\}$$

- Le **poids minimale** d'un code

$$\boldsymbol{w}(C) = min\{\boldsymbol{w}(x) : x \in C, x \neq 0\}$$

Exemple Pour le code de l'exemple precedent on a :

$$d(x, y) = d(01110, 10101) = 4$$

$$d(x, z) = d(01110, 11011) = 3$$

$$d(y, z) = d(10101, 11011) = 3$$

La distance minimale de C est $d(C) = 3$ et son poids minimale est $\boldsymbol{w}(C) = 3$

La proposition suivante due à Hamming nous donne une borne sur le nombre de mots du code

Proposition Soit $card(B) = 2$ et $C \subset B^n$ un code qui corrige jusqu'à e erreurs. Alors

$$card(C) \leq \frac{2^n}{\sum_{i=0}^{e} \binom{n}{i}}$$

La notion de **rayon de recouvrement** spermet de mesurer à quel point un mot reçu z peut différer d'un mot du code $c \in C$. Le rayon de recouvrement est défini par

$$\rho = max\{min\{d(x,c) :\in C\}, z \in B^n\}$$

Le rayon de recouvrement est le plus petit ρ pour lequel les boules $B_\rho(c)$ pour $c \in C$ recouvrent l'ensemble B^n en entier.

Soit t le plus grand entier tel que les boules $B_t(c)$, $c \in C$ soient disjointes. Si $\rho = t$, on dit que le code est **parfait**. En d'autre termes, on dit qu'un code $C \subset B^n$ de distance minimale $h(C) = 2e + 1$ est parfait si tout mot x de B^n est à la distance $\leq e$ d'exactement un mot c du code.

Condition d'empilement des sphères :

Si $C \subset B^n$ est un code parfait corrigeant e erreurs, alors pour $card(B) = q$

$$card(B) \sum_{i=0}^{e} \binom{n}{i} (q-1)^i = q^n$$

Problème du décodage

Détection

On suppose avoir reçu un message qui n'est pas un mot du code. Il est clair qu'il y a eu une erreur au cours de la transmission et nous avons détecté la présence d'une (ou de plusieurs) erreur. Si aucune erreur n'a été détectée, on a

- soit reçu un mot du code
- soit reçu un mot qui comportait trop d'erreurs et, dans ce dernier cas, le code n'était pas adapté à la capacité du canal

Correction

On va supposer que le mot à corriger doit être le plus proche possible d'un mot correct (d'un mot du code).

Exemple Soit

$$C_1 = \{00, 01, 10, 11\}$$

Chaque mot reçu est un mot du code. C_1 ne peut donc servir à détecter des erreurs. C_1 ne corrige pas d'erreurs non plus.

Exemple On modifie C_1 en répétant trois fois chaque mot du code:

$$C_2 = \{00\,00\,00, 01\,01\,01, 10\,10\,10, 11\,11\,11\}$$

Ce code s'appelle code à répétition. Supposons avoir reçu 110101. Il ne s'agit pas d'un mot du code et on peut affirmer qu'au moins une erreur est apparue. En ne changeant qu'un seul bit, on peut former le mot du code 010101 mais on peut également obtenir d'autres mots du code en changeant plus d'un bit. On suppose donc que le mot du code correct est 010101 et on corrige donc 110101 en 010101.

Exemple On modifie C_1, le code de l'exemple précédent, en ajoutant un troisième bit à chaque mot de façon à ce que le nombre de 1 des mots soit pair:

$$C_3 = \{000, 011, 101, 110\}$$

Le bit ajouté s'appelle **bit de parité**. Supposons avoir reçu le message 010. Comme 010 n'est pas un mot du code, on est certain qu'il y a eu une erreur de transmission. Le message peut être décodé en

- 110,
- 000
- 011

en ne changeant qu'un seul bit du message. Nous allons distinguer la manière de traiter les mots reçus les plus proches d'un seul mot du code.

Codes linéaires

Préambule mathématique

On dit que E est un espace vectoriel sur un corps K si et seulement si, pour des éléments u, v et w de E, on a :

1. $(u + v) + w = u + (v + w)$
2. $\exists 0 : u + 0 = 0 + u = u$
3. $\forall u \exists (-u) : u - u = 0$
4. $u + v = v + u$
5. $\forall c \in K, c(u + v) = c \cdot u + c \cdot v$
6. $\forall a, b \in K, (a + b) \cdot u = a \cdot u + b \cdot v$
7. $\forall a, b \in K, (a \cdot b)u = a \cdot (b \cdot u)$
8. $1 \cdot u = u$

Soit E un espace vectoriel sur un corps K.

$F \neq \emptyset$ est un sous-espace vectoriel de E si et seulement si :

- $x + y \in F \quad \forall x, y \in F$
- $\lambda x \in F \quad \forall \lambda \in K, \forall x \in F$

Si A est un sous-ensemble de E, alors le sous-espace engendré par A est l'ensemble de toutes les combinaisons linéaires d'éléments de A.

Une **famille génératrice** de E est un sous-ensemble $G \subset E$ tel que le sousespace engendré par G est E. Ou, de manière équivalente, que tout vecteur de E est une combinaison linéaire d'éléments de G. E est dit de dimension finie s'il contient une famille génératrice finie.

Une **famille libre** de E est un sous-ensemble $L \subset E$ tel qu'aucun élément $v \in L$ n'est une combinaison linéaire d'autres éléments de L. Ou, de manière équivalente, la seule combinaison linéaire d'éléments de L qui est nulle est celle dont tous les coefficients sont nuls.

Si la dimension de E est finie alors chaque base de E est finie (son ensemble sous-jacent est de cardinal fini) et toutes les bases ont le même nombre d'éléments. Ce nombre est la dimension de l'espace.

Si la dimension de E est n, alors:

- Si $\{g_i\}_{i\in I}$ est une famille génératrice de E alors $card(I) \geq n$. S'il y a égalité, alors $\{g_i\}_{i\in I}$ est une base
- Si $\{\ell_i\}_{i\in I}$ est une famille libre de E alors $card(I) \leq n$. S'il y a égalité, alors $\{\ell_i\}_{i\in I}$ est une base.
- Si $E = K^n$, alors la famille $\{e_i\}_{i\in I}$ avec $I = \{1, 2, ..., n\}$ et $e_i = (0, 0, ..., 1, 0, ..., 0)$ avec 1 à la $i$eme position est la base canonique de K^n.

Soient E et V deux espaces vectoriels sur un corps K. Une application linéaire f de E dans V vérifie:

- $\forall x \in E, \forall y \in E \quad f(x+y) = f(x) + f(y)$
- $\forall \lambda \in K, \forall x \in E \quad f(\lambda \cdot x) = \lambda \cdot f(x)$

Définition Un code linéaire C de longueur n sur un alphabet à 2 lettres qu'on identifie à $Z_2 = \{0, 1\}$ est un sous-espace linéaire de l'espace vectoriel $(Z_2)^n$. Autrement dit C vérifie :

- $\forall u, v \in C, \ u + v \in C$
- $\forall u \in C, \forall a \in Z_2, a \cdot u \in C$
- Si C a pour dimension k (au sens des espaces vectoriels), C est appelé (n, k)-code linéaire
- Si sa distance minimale est d, C sera appelé un (n, k, d)-code linéaire.

Définition On définit le taux d'information du code C de longueur n comme le rapport

$$R = \frac{1}{n} \cdot log(card(C)) = \frac{k}{n}$$

Le code C peut être défini au moyen d'une matrice G à k lignes et n colonnes appelée matrice génératrice dont les lignes forment une base de C. Soient $\{v_1, v_2, \ldots, v_k\}$ les vecteurs lignes de G. Tout élément x de C peut être exprimé comme une unique combinaison linéaire de ces lignes :

$$x = \sum_{i=1}^{n} a_i \cdot v_i$$

pour des $a_i \in Z_2$. G est une application linéaire

$$(Z_2)^k \rightarrow (Z_2)^n$$

qui associe à tout mot de longueur k sur l'alphabet Z_2 (un vecteur de $(Z_2)^k$), un mot de longueur n sur Z_2, (un vecteur de $(Z_2)^n$).

On ajoute de cette manière $n - k$ symboles de redondance aux mots binaires à k lettres.

A partir des vecteurs de $(Z_2)^k$ et de la matrice G, on peut énumérer les éléments de C :

$$C = \{a \cdot G : a \in (Z_2)^k\}$$

Exemple Soit la matrice génératrice d'un code C.

$$G = \begin{pmatrix} 011 \\ 100 \end{pmatrix}$$

Les mots du code sont :

	$\begin{pmatrix} 011 \\ 100 \end{pmatrix}$
00	000
01	100
10	011
11	111

- Le code de dimension 2
- possède 4 mots $\{000, 100, 011, 111\}$
- Il est de distance minimale 1
- son taux d'information est $\frac{1}{3} \cdot log_2(4) = \frac{2}{3}$

On dira que la matrice génératrice G est sous forme normale si G est de la forme :

$$A = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 & p_{1,1} & p_{1,2} & \cdots & p_{1,n-k} \\ 0 & 1 & \dots & 0 & 0 & p_{2,1} & p_{2,2} & \cdots & p_{2,n-k} \\ \dots & & & & & & & & \\ 0 & 0 & \dots & 0 & 1 & p_{k,1} & p_{k,2} & \cdots & p_{k,n-k} \end{pmatrix}$$

Dans ce cas, les k premiers symboles d'un mot de C sont appelés les **symboles d'information** et les $n-k$ autres les **symboles de redondance**

On dit que deux (n,k)-codes C et C' sont équivalents si C' peut être obtenu à partir de C en appliquant une permutation donnée aux lettres de tous les mots de C

Deux $k \times n$ matrices G et G' engendrent des (n, k)-codes linéaires équivalents si on peut obtenir G à partir G' par une suite d'opérations à choisir parmi:

- permutation des lignes;
- addition de deux lignes;
- permutation des colonnes.

Théorème La distance minimale d'un code linéaire est égale à son poids minimal

Décodage par les classes latérales

Dans la suite, C désignera toujours un (n, k)-code linéaire sur Z_2, =un sous-espace de dimension k de $(Z_2)^n$

Supposons que le mot du code $x = x_1 x_2 \dots x_n$ est émis au travers d'un canal bruité et que le mot reçu est $y = y_1 y_2 \dots y_n$

Définition Le vecteur d'erreur $e = y - x$

Étant donné y, le décodeur doit décider quel mot du code x a été transmis, ou de manière équivalente, quel est le vecteur d'erreur. Les codes linéaires fournissent une solution élégante au problème du décodage au plus proche voisin de manière à minimiser le poids de l'erreur- , au moyen des classes latérales

Définition Pour un code C et un vecteur $u \in (Z_2)^n$, on appelle **classe latérale** de C l'ensemble $u + C$ défini par:

$$u + C = \{u + x : x \in C\}$$

Exemple Si $C = \{0000, 0101, 1011, 1110\}$ alors :

- $0000 + C = C$ lui-même
- $1000 + C = \{1000, 1101, 0011, 0110\}$
- $0100 + C = \{0100, 0001, 1111, 1010\}$
- $0010 + C = \{0010, 0111, 1001, 1100\}$
- $0001 + C = \{0001, 0100, 1010, 1111\}$

Observons que $0001 + C = 0100 + C$

Lemme Soit $u + C$ une classe latérale de C. Si $v \in u + C$, alors $v + C = u + C$

Le théorème suivant affirme que les classes latérales permettent de partitionner l'ensemble de tous les mots possibles sans qu'il y ait de recouvrement entre les classes

Théorème Soit C un (n, k)-code linéaire sur Z_2, alors

1. tout vecteur de $(Z_2)^n$ est dans une classe latérale de C
2. chaque classe latérale contient exactement 2^k vecteurs
3. étant donné deux classes latérales, elles sont soit disjointes soit identiques

Décodage par les classes latérales

On partitionne $(Z_2)^n$ en

$$(0 + C) \cup (u_1 + C) \cup \ldots \cup (u_s + C)$$

où $s = 2^{n-k} - 1$ et où les $0, u_1, \ldots u_s$ sont des éléments de poids minimal appelés **chefs de classe**.

On peut alors construire le tableau standard de C qui est une matrice à 2^{n-k} lignes et 2^k colonnes. Il contient tous les vecteurs de $(Z_2)^n$.

Sa première ligne correspond aux mots de C avec le vecteur 0 à gauche; les autres lignes représentent les classes latérales $u_i + C$ avec leur chef de classe à gauche. L'algorithme suivant permet de construire le tableau standard:

1. on énumère les mots de C en commençant par 0 sur la première ligne;
2. on choisit un vecteur u_1 de poids minimal qui n'apparaît pas dans la première ligne et on énumère sur la deuxième ligne les éléments $u_1 + C$ en inscrivant au-dessous de 0 le chef de classe u_1 et au-dessous de chaque élément $x \in C$ l'élément $u_1 + x$;
3. on choisit un vecteur u_2 de poids minimal qui n'apparaît pas dans les premières lignes et on énumère sur la troisième ligne les éléments $u_2 + C$ en inscrivant au-dessous de 0 le chef de classe u_2 et au-dessous de chaque élément $x \in C$ l'élément $u_2 + x$;
4. on itère ce procédé jusqu'à ce que toutes les classes latérales soient listées et que tout vecteur de $(Z_2)^n$ n'apparaisse qu'une seule fois.

Le décodeur va utiliser le tableau standard de la façon suivante: lorsque le mot y est reçu, on recherche sa position dans le tableau standard. Le décodeur décide alors

- que le vecteur d'erreur e correspond au chef de classe qui est situé dans la première colonne de la même ligne et
- peut décoder y comme $x = y - e$ en choisissant le mot du code de la première ligne sur la même colonne que y.

Les vecteurs d'erreurs qui pourront être corrigés sont précisément les chefs de classe, quel que soit le mot du code transmis. En choisissant des vecteurs d'erreur de poids minimal en tant que chefs de classe, le tableau standard assure un décodage au plus proche voisin.

Observons cependant que ce procédé de décodage est trop lent pour de grands codes et trop coûteux en termes de mémoire. En effet, si le nombre d'éléments du tableau est $2^{n-k} \cdot 2^k = 2^n$, la complexité de cet algorithme est $O(2^n)$. L'algorithme est exponentiel, ce qui est impropre au décodage. Il faut de surcroît mémoriser la totalité de la table, ce qui implique un coût mémoire également exponentiel. Il existe un autre moyen moins inefficace de décoder.

Exemple

On cherche à transmettre des messages α de longueur 2 sur l'alphabet $\{0,1\}$ au moyen du $(4,2)$-code linéaire défini par la matrice génératrice G suivante:

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \tag{4}$$

On observe que G n'est pas sous forme standard. On transforme alors G et on obtient la matrice génératrice G' d'un code équivalent qui est sous forme standard:

$$G' = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix} \tag{5}$$

L'ensemble des messages non codés correspond aux différents couples possibles sur $\{0,1\}$ qui sont:

$$A = \{(0,0),(0,1),(1,0),(1,1)\}$$

On peut alors énumérer les différents mots du code C en effectuant le produit à gauche des éléments $\alpha \in A$ par la matrice G' :

mots de A	$\begin{pmatrix} 1011 \\ 0101 \end{pmatrix}$	poids
00	0000	
01	0101	2
10	1011	3
11	1110	3

Le code C est donc composé des mots:

$$C = \{0000, 0101, 1011, 1110\}$$

Le poids minimal des mots de C nous donne la distance minimale du code qui est 2. Le code C est donc un $(4, 2, 2)$-code linéaire.

Si on veut transmettre le message (1,0), il suffit d'effectuer le produit

$$\begin{pmatrix} 10 \end{pmatrix} \cdot \begin{pmatrix} 1011 \\ 0101 \end{pmatrix} = (10\,11)$$

Avec

- 10 symboles d'information
- 11 bits de redondance

Afin de corriger une erreur, le décodeur construit le tableau standard suivant:

0000	0101	1011	1110	$\leftarrow$ mots de C
1000	1101	0011	0110	
0100	0001	**1111**	1010	
0010	0111	1001	1100	

$\uparrow$

chef de classe

obtenu à l'aide des classes latérales:

$0100 + C$ puisque $0001 \in 0100 +$ C.

- $0000 + C = C$ lui-même
- $1000 + C = \{1000, 1101, 0011, 0110\}$
- $0100 + C = \{0100, 0001, 1111, 1010\}$
- $0010 + C = \{0010, 0111, 1001, 1100\}$
- $0001 + C = \{0001, 0100, 1010, 1111\}$

Observons que la classe latérale $0001 + C$ est identique à la classe latérale $0100 + C$ puisque $0001 \in 0100 + C$.

Si on suppose avoir reçu le message 1111, on vérifie facilement que ce n'est pas un mot du code. Pour trouver de quel mot du code il provient :

- on cherche sa position dans le tableau standard et on lit le mot du code qui est dans la même colonne sur la première ligne
- le vecteur d'erreur se lit sur la même ligne dans la première colonne.

Ainsi, le message transmis était 1011 avec 0100 comme vecteur d'erreur.

Remarque Le code de cet exemple peut corriger une erreur si celle-ci se rencontre sur une des trois premières positions du mot mais pas dans la quatrième. Par exemple, le message 01 est codé en 0101 et altéré en 0001. On le décode convenablement.

En revanche, le même message altéré sur sa dernière position donne 0100 qui est décodé improprement en 00. On retrouve ainsi le fait que, comme $d(C) = 2$

C n'est pas un code correcteur, mais seulement détecteur d'une erreur.